DISSERTATION

Visually and Physically Plausible Object Pose Estimation for Robot Vision

conducted in partial fulfillment of the requirements for the degree of a

Doktor der technischen Wissenschaften (Dr. techn.)

supervised by

Ao.Univ. Prof. Dipl.-Ing. Dr. techn. Markus Vincze
E376 Automation and Control Institute

submitted at the

TU Wien
Faculty of Electrical Engineering and Information Technology

by

Dipl.-Ing. Dominik Bauer
DOB 08.05.1992
Matr. Nr.: 1227005

Vienna, December 2021 Dominik Bauer

Acknowledgment

This thesis and the work therein would not have been possible without the people I have been fortunate to be surrounded with over the years.

I would like to thank Markus Vincze for providing me with this opportunity and always having an open door for discussion. It has been my pleasure to collaborate with Timothy Patten, who helped me shape my ideas through his mentorship and commitment. I am happy to have had such wonderful colleagues that created a great atmosphere, in the lab and during our time off, never shy to offer a helping hand or share a laughter. The doctoral college TrustRobots enabled a motivating start into this journey, made possible by all the cordial people involved who have accompanied me since.

To all the anonymous reviewers whose feedback allowed to improve my work and motivated me to come up with new ideas. I am also grateful to have received funding by TU Wien in the course of the doctoral college, the Austrian Science Fund (FWF) under grant agreement No. I3968-N30 HEAP and the OMRON Corporation.

My personal thanks go to all my friends, no matter if I had the chance to see them every day or just a few times a year. None of this would have been possible without the support of my family and especially my parents, Günther and Monika, who always had my back. And a heartfelt thank you to my partner Natalie for her warmth and understanding, for supporting me in stressful times and for helping me to take my mind off things.

I am lucky to have everyone of you around me.

Abstract

Autonomous robots are expected to reliably interact with their environment, following user commands and manipulating objects. This requires a robot to understand its environment, to determine the objects of which it is composed and how they relate to each other. Using object pose estimation, the robot may determine the 3D translation and 3D rotation of known object models with respect to its observation of the environment. Given the pose of all observed objects, the robot may create a 3D representation of the scene, consisting of the objects' models and the spatial relations between them. Such an understanding allows the robot to, for example, reason about interactions with individual objects, synthesize novel views of the scene or interpret users' commands. However, the alignment of object models to the robot's visual observation may suffer from sensor noise, partial observability and object symmetry that lead to ambiguous situations and inaccurate poses. The resulting representation of the scene may thus contain implausibilities such as intersecting, floating or statically unstable objects. Resorting to physical relations alone also suffers from ambiguity as there are, for example, numerous possibilities for two objects to plausibly interact. Accounting for such scene-level consistency is further complicated by multiple, potentially inaccurate hypotheses per object that create a complex search space for resolving conflicting pose hypotheses.

To overcome these ambiguities and to resolve scene-level inconsistencies, we hypothesize that visual and physical plausibility complement each other and allow for more accurate and robust object pose estimation. We conjecture that the complexity of dealing with scenes of multiple objects with multiple hypotheses each may be tamed by considering the plausibility of the resulting configurations. While we argue that such reasoning may be generally beneficial in robot vision, we focus on the task of object pose estimation and its sub-steps of refinement and verification. In this thesis, we provide definitions for visual and physical plausibility of object poses in static scenes. Visual plausibility is considered as rendering- or point cloud-based alignment. Physical plausibility is determined by simulation or evaluation of static equilibrium. We propose analytical and learning-based approaches to the object pose estimation task that leverage these definitions. We explore concepts from reinforcement learning to incorporate plausibility at different stages of the pose estimation pipeline and to efficiently consider vast numbers scene-level combinations. Moreover, based on the plausibility information gathered by our proposed methods, we derive explanation strategies for human-robot interaction in case of robotic failure. By evaluation on common datasets and by applying our methods to robotic grasping, we highlight the accuracy, robustness and efficiency of our proposed object pose estimation approaches and demonstrate the benefit of considering visual and physical plausibility for this task.

Contents

Chapter 1

Introduction

A robot's ability to explain its own actions – or reasons for why they might have failed – is an important building block for establishing and maintaining human trust [1]–[3]. But to provide such explanations, the robot is required to attain a thorough understanding of the scene it inhabits. This understanding may include the scene's objects, their location and their relations to one another [4] – expressed as their class, the associated 3D model, its pose and spatial relations. A scene understanding represented by 3D models allows to derive information about the scene that enables a robot to explain and carry out interactions with its environment, as illustrated in Figure 1.1.

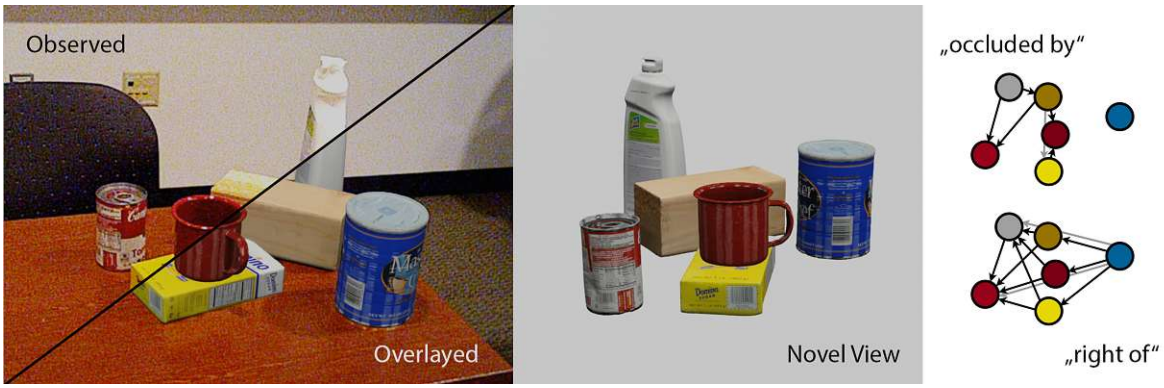


Figure 1.1: 3D model-based scene understanding. An observation with overlayed 3D models (left), a novel synthesized view of the scene (mid) and derived view-dependent relations between the objects (right), visualized as directed graphs.

For example, by rendering the estimated scene and comparing it to its camera image, a robot may determine the plausibility of its scene understanding. Spatial relations between objects in the scene may be derived, allowing the robot to reason about what could be done with the objects. If a mug is upright, something may be put inside – if it is upside-down, something may be stacked upon it. Such relations also allow the robot to make sense of human requests such as "hand me the can to the right of the mug".

Moreover, this understanding enables the robot to carry out tasks such as grasping and manipulating objects [5]–[7], determining where to grasp an object and in which order to manipulate scene objects without causing other objects to fall over.

An exemplary robotic pipeline for this task is shown in Figure 1.2. To gather an understanding of the observed scene, the robot locates the known objects in the observed image – the object pose estimation task. Thereby, the robot may determine the relative pose between the object and itself, plan how to grasp it and eventually execute its plan to manipulate the scene.

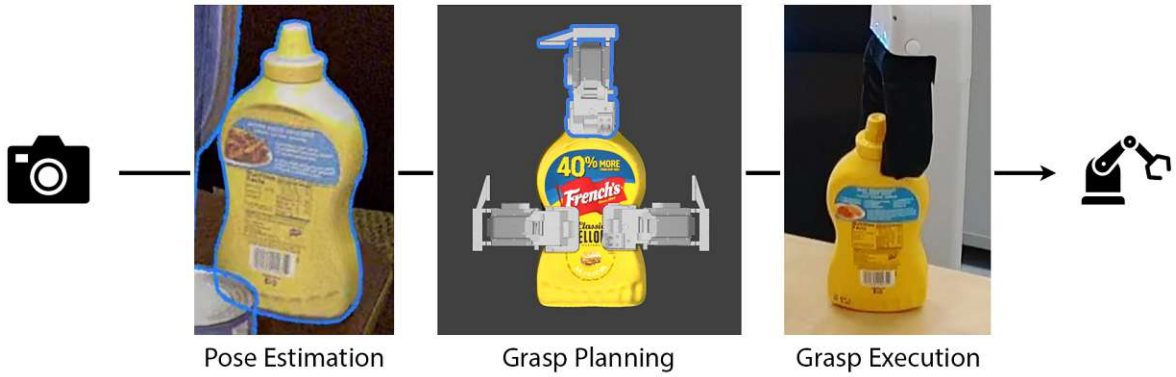


Figure 1.2: Application example: A grasping pipeline. The robot perceives the object of interest, estimates its pose, selects the best grasp and finally executes it.

The focus of the presented work is on the first step in this pipeline: object pose estimation. This step consists of generating initial pose hypotheses, their refinement and verification to determine an accurate pose estimate. Furthermore, we consider their application to robotic grasping from a robot vision and a human-robot interaction (HRI) perspective.

1.1 Task: Object Pose Estimationn

We assume both color and depth information to be available – commonly provided by the robot's RGBD camera – and the 3D models of the objects of interest to be given. In addition, the 2D detections and instance segmentation masks are given, indicating which area of the observation contains the target object. But to plan a semantically meaningful grasp like grasping a mug by its handle, the observation must be related to the actual object model.

Object Pose Estimation: As illustrated in Figure 1.3, the task of pose estimation is to align the 3D model with the observation of the object. Thereby, the 3D translation and 3D orientation of the object model in the observation is determined. The combination of these transformations is called the *6D pose* of the object with respect to the camera. Note that this may be done using only color [8], [9], only depth [10], [11] or both modalities [12], [13].

A challenge is that a small error with respect to the 2D color information may result in a large offset in the camera's viewing direction. As a result, the robot may not be able to reliably grasp the object. The additional use of depth information allows to determine an object's distance from the camera. The 2D information may be lifted to a 3D point cloud, as illustrated in Figure 1.3. But the resulting point cloud may suffer from noise and missing values, shown as black areas in the depth image in Figure 1.3. Moreover, misdetections or severely inaccurate pose estimates may result in the robot failing to grasp or even colliding with its environment.

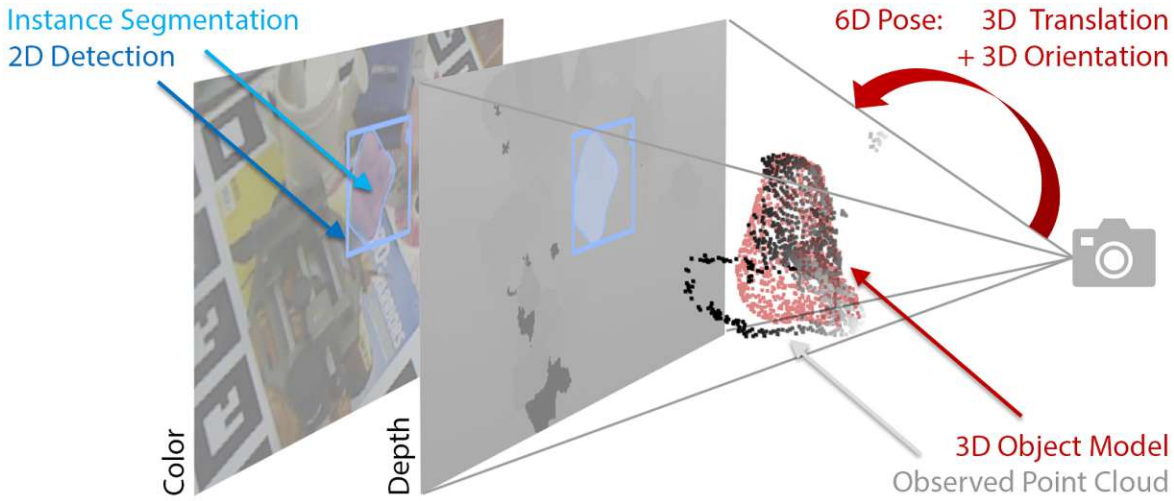


Figure 1.3: Given a color image, we compute a detection that locates an object of known class in the 2D image and an instance segmentation that assigns individual pixels to this class (or the background). Transferred to an observed depth image, a point cloud may be generated that represents the known object. The task of object pose estimation is then to determine the 6D pose of a 3D model (corresponding to the detected object) such that it visually aligns to the observed image or point cloud.

Due to these issues, object pose estimation (for robot vision) typically also includes a refinement step that improves the accuracy of poses and a verification step that aims to ensure that the object under estimated pose actually aligns with the observation.

Object Pose Refinement: Starting from a potentially inaccurate initial pose estimate, the task of refinement is to improve the alignment of the 3D model to the observed object. This might be done using color information [14], [15] or using both color and depth modalities [12], [16]. Pose refinement may be considered as mask alignment in 2D or incorporate higher-dimensional feature matching. Most commonly, however, variants of the ICP algorithm are used [17], [18] that alternate between finding correspondences between a model point cloud and the observed point cloud, iteratively aligning both more closely. Similar to the initial pose estimation, the refinement process may not yield an accurate pose. The refinement may even diverge from the true object pose in the worst case.

Hypotheses Verification: As both detection and pose estimation methods may yield multiple hypotheses that are potentially inaccurate or even represent a misdetection, verification methods are employed as a means to explicitly check the alignment of given pose hypotheses. Thereby, the hypothesis that is best supported by the observation may be determined, allowing to increase the robustness of the overall pipeline as compared to one-shot approaches. Different scores are proposed to this end, based on depth images [13], [19], point clouds [20], [21] and even considering geometrical or physical interactions between the estimated scene objects [22], [23].

1.2 Challenges and Research Questions

The challenges shown in Figure 1.4 affect different parts of the discussed perception pipeline – from detection and segmentation to pose estimation, refinement and verification. As a result, the robot's scene understanding might end-up with multiple inaccurate pose hypotheses.

Visual Alignment The observation in the form of RGB or depth images is only a partial and noise afflicted view of the object, due to limited visibility from a single view and sensor limitations. Symmetrical objects may also lead to ambiguity as the object would align to the same observation under multiple poses. Such symmetry ambiguity may also be introduced by (self-)occlusion, hiding discriminatory parts of the object. Moreover, such occlusions only allow partial observation of the object of interest, increasing the visibility challenge. A further source of partial observability is inaccurate segmentation, pruning too much of the foreground or falsely including background points.

Physical Relations The inaccuracy resulting from these visual ambiguities may also be observed in the estimated objects' physical relations. Objects under inaccurate pose intersect each other, float in the air or would not remain statically stable. However, the physically plausible configurations of an object are ambiguous too – an object may rest on a plane in any of its stable poses, for example. Considering the physical configuration alone may thus result in divergence from the observation. In addition, information about the scene's support and the gravity direction may be inaccurate, propagating to the estimated scene.

Scene-level Consistency Estimating each objects' pose individually will therefore not guarantee a valid scene configuration. For hypotheses verification, the potentially large search space of combinations of multiple interacting objects under multiple poses and the computational load of determining the verification scores for each of them, however, quickly becomes intractable.

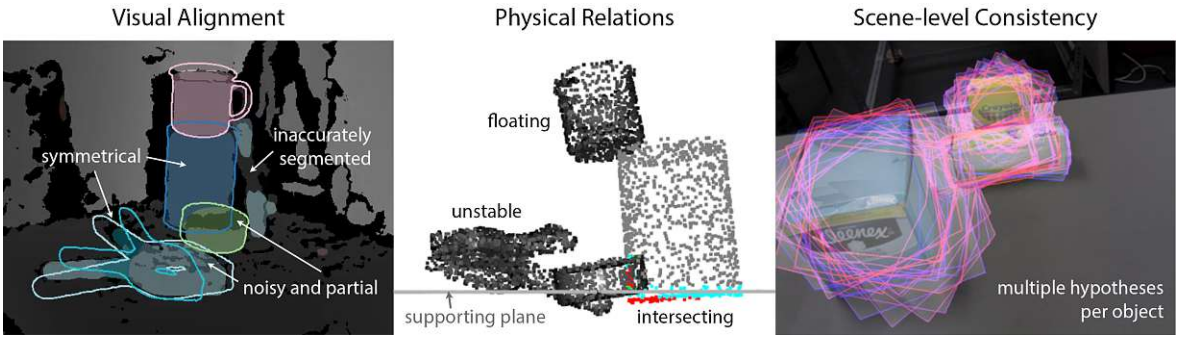


Figure 1.4: Challenges in object pose estimation. Limited visibility, noise and inaccurate segmentation result in inaccurate pose estimates in clutter scenes (left). The physical object relations in the resulting scenes violate the assumptions of plausible, static scenes (mid). However, the complexity of considering all scene-level interactions of multiple object under multiple (inaccurate) pose hypotheses each quickly grows intractable (right).

We conjecture that both the visual and physical plausibility of the robot's scene understanding need to be jointly considered to resolve ambiguities in the observation. To this end, we require a definition of plausibility that may be incorporated with the steps outlined for pose

estimation. To tame the complexity of accounting for scene-level interactions, we need to efficiently decide on which hypotheses to prune, refine and select as final estimate. Additionally we hypothesize that, for the robot to provide an effective explanation of its understanding of a scene and its interactions with it, it needs to resolve to human-understandable reasoning approaches – such as how well the robot's understanding visually aligns with its camera images or how physically plausible the pose of an object would be in a simulation.

These assumptions give rise to the following research questions to be investigated:

Research Questions

- How to define the plausibility of object poses in static scenes?
- How to overcome the ambiguity of using either visual or physical reasoning in object pose estimation, refinement and verification?
- How to efficiently consider multiple objects with multiple hypotheses?
- How to leverage the plausibility information for explanation of robotic failure?

1.3 Contributions and Outline

In addressing these research questions, we propose definitions of visual and physical plausibility for static scenes or rigid objects. We leverage these definitions in object pose estimation, refinement and verification and propose a set of simulation and learning-based methods, respectively. By consideration of plausibility, our proposed methods are able to accurately estimate objects' poses, while exposing robustness to observational noise and inaccurate initialization. We investigate reinforcement learning concepts that enable efficient exploration of vast hypotheses spaces and allow to incorporate plausibility considerations as reward functions. Furthermore, we derive visualizations and textual explanations for use in HRI and propose a user study design that evaluates their effect on the robot's trustworthiness and understandability in case of failure.

Contributions

- Definition of visual and physical plausibility of object poses using rendering, simulation and contact-based formulations
- Exploitation of these principles for object pose estimation, refinement and verification – increasing accuracy, robustness and speed over competing approaches
- Proposal of simulation and learning-based methods – exploring different reinforcement learning approaches to incorporate plausibility considerations
- Leveraging the plausibility information for robotic grasping and explanation of its failure in HRI scenarios – increasing understandability and trust

In the following, we provide an overview of this thesis and relate the contributions to the publications listed in Section 1.5.

1.3.1 Defining Visual and Physical Plausibility of Object Poses

Visual alignment alone may lead to ambiguity under partial observability. We argue that physical plausibility is able to resolve visually ambiguous cases – and vice-versa.

Rendering-based Visual Plausibility: We define a visual-alignment score [24] that quantifies the average alignment between the object in the observed (i.e., under ground-truth pose) and rendered depth and normal images under the estimated pose.

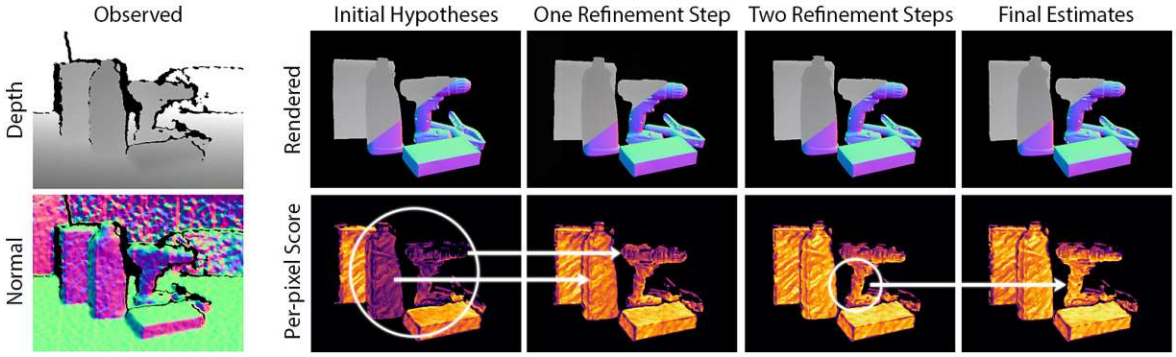


Figure 1.5: Definition of visual plausibility based on rendering-based scoring. The observation (left) is compared to the rendered objects under estimated pose (right). The resulting score is visualized below (more yellow indicates more aligned). [25].

Contact- and Simulation-based Physical Plausibility: We define the physical plausibility of a scene [26] as the combination of feasibility (non-intersecting, non-floating) and static stability of the objects therein, as illustrated in Figure 1.6. Instead of determining the physical plausibility based on contact points, we may also initialize the estimated scene in a physics simulation and evaluate its dynamic progression over time. To determine a stable pose, for example, we may want to simulate until an object remains static.

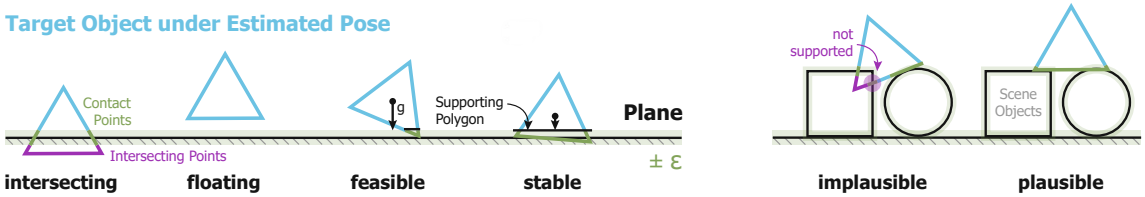


Figure 1.6: Definition of physical plausibility based on contact points. A plausible pose results in a non-intersecting, non-floating and stable configuration. [25].

1.3.2 Enforcing Plausibility through Rendering and Simulation

By using rendering and physics simulation, illustrated in Figure 1.7, 3D models are all that is required by the methods presented in this section to consider new objects. The proposed approaches enforce plausibility, exploit it to limit the search space given multiple pose hypotheses and to improve initial poses.

Stable Object Pose Estimation: The predictive power of considering plausibility for this task is shown by a proof-of-concept pose estimator in [27]. It assumes only approximate object meshes and segmentation masks to be given; no additional training is required for pose estimation. Through physics simulation and clustering, we derive a small set of physically plausible poses per object. Using a rendering-based scoring, we are able to determine the visually most plausible candidate.

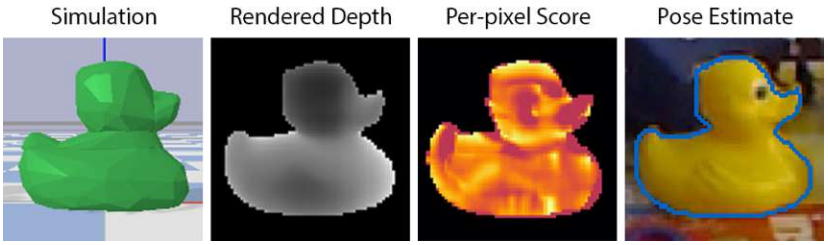


Figure 1.7: Physically plausible pose hypotheses (simulation) are evaluated using a rendering-based scoring, yielding the overall most plausible as pose estimate. [27].

Integrated Object Pose Refinement and Verification: The method presented in [24] integrates iterative refinement, physics simulation and rendering-based scoring in a joint optimization. Thereby, simulation guides refinement towards physically more plausible poses, while alignment-based refinement improves visual plausibility. To contain divergence, we embed rendering-based scoring in the refinement loop. Handling a large number of hypotheses, we propose to consider the allocation of a fixed refinement budget as a MAB problem, balancing exploration of the different hypotheses with exploitation of high-scoring hypotheses. We extend our approach towards multiple objects, clustering objects by their scene-level interactions. We iteratively add objects from the ordered clusters to the simulated scene, allowing consideration of occlusions and support relationships between them.

1.3.3 Enforcing Plausibility in Learning-based Approaches

The methods just presented consider the visual and physical aspects of plausibility in isolation. However, improving physical plausibility through simulation competes with improving visual plausibility through iterative refinement. Instead, the influence of both plausibility aspects should be dynamically adapted depending on the scene configuration and refinement state. This motivates the design of a learning-based, plausible pose refinement approach.

Reinforced Point Cloud Registration: As a first step in this direction, in [28], we propose a novel approach to the related task of point cloud registration. We pose the iterative registration task as determining a policy that selects basic registration actions in each step. Our registration agent is trained using a combination of IL and RL. Through IL, the agent should learn to replicate the behavior of an expert that has perfect information. In addition, the agent is guided by a symmetry-aware point-cloud alignment reward. This formulation allows to integrate further constraints by including them in the agent's reward.

Reinforced Object Pose Refinement and Verification: When applying this registration method to cluttered scenes such as the ones observed in Figure 1.1, additional consideration of contact-based physical plausibility allows to resolve the resulting visual ambiguities.

In [25], we reward the agent when it attains a physically plausible scene configuration. The objects' surface distance provides the underlying information to determine plausibility and, furthermore, situates the objects within the scene. We adapt the expert policy to account for symmetrical objects by following the shortest trajectory to any symmetrical pose. The introduction of an outlier-removal subnetwork and rendering-based verification of the refinement steps robustifies the method in cluttered scenes. Thereby, a learning-based approach is achieved that jointly considers both aspects of plausibility.

1.3.4 Explaining Plausibility Violations

The consideration of plausibility not only serves a technical benefit but also supports users' understanding of the robot's perception and actions and, thereby, fosters trust. In [29], we investigate how plausibility-based explanations of robotic failure are perceived by human interaction partners. We propose multi-modal (textual and visual) explanations, presented in a single-shot manner or interactively – illustrated in Figure 1.8. Our proposed online study evaluates the impact of different explanation strategies on users' understanding of the robot and its perceived trustworthiness after the interaction.

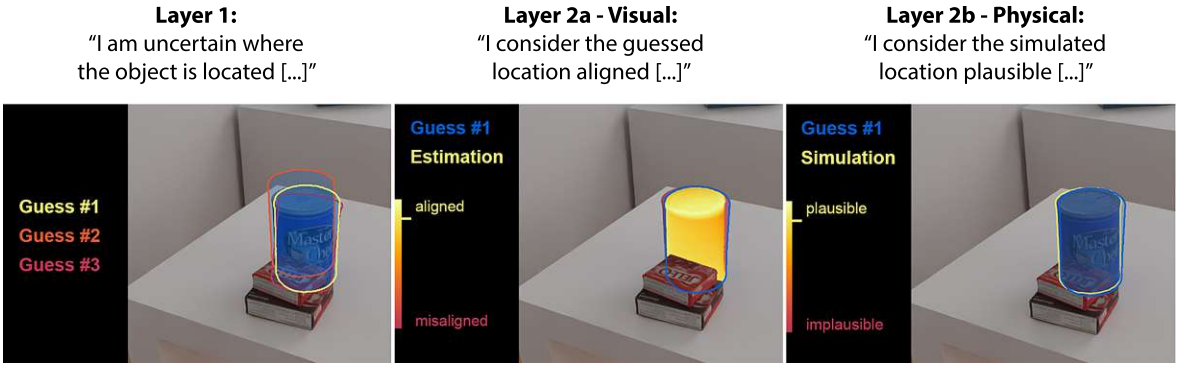


Figure 1.8: Multi-modal explanation of robotic failure. Exemplary textual explanations are shown together with visualizations of uncertainty (left), visual plausibility (mid) and physical plausibility (right). Users interactively request more detailed explanation layers in case of robotic failure. [29].

1.4 Organization

We propose definitions of visual plausibility through rendering and physical plausibility by simulation or evaluation of the static equilibrium in Chapter 3. We present approaches to exploit these plausibility definitions in object pose estimation. The methods we propose in Chapters 4 and 5 leverage rendering and simulation in the context of Multi-armed Bandits (MAB). In Chapters 6 and 7, we propose novel object pose refinement methods based on Imitation and Reinforcement Learning (IL and RL) that jointly consider both presented aspects of plausibility. Finally, we present reasoning strategies that exploit this information for explanations in HRI and an accompanying user study design in Chapter 8.

1.5 List of Publications

The contents of this thesis were previously presented in the following publications:

- **Bauer, D.**, Patten, T., & Vincze, M. (2022). Visual and Physical Plausibility of Object Poses for Robotic Scene Understanding. *Under review.*
- **Bauer, D.**, Patten, T., & Vincze, M. (2022). SporeAgent: Reinforced Scene-level Plausibility for Object Pose Refinement. *IEEE Winter Conference on Applications of Computer Vision (WACV), 196-204*. DOI: 10.1109/WACV51458.2022.00027
- **Bauer, D.**, Patten, T., & Vincze, M. (2021). ReAgent: Point Cloud Registration using Imitation and Reinforcement Learning. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 14586-14594*. DOI: 10.1109/CVPR46437.2021.01435
- Papagni, G.*, **Bauer, D.***, Köszegi, S., & Vincze, M. (2021). A Study Design for Evaluation of Trust and Understandability through Interactive Multi-Modal Explanations of Robotic Failure. *HRI 2021 Workshop Your Study Design.*
- Vincze, M., Patten, T., Park, K., & **Bauer, D.** (2020). Learn, detect, and grasp objects in real-world settings. *Elektrotechnik und Informationstechnik (e&i)*, 137(6), 324-330. DOI: 10.1007/s00502-020-00817-6
- **Bauer, D.**, Patten, T., & Vincze, M. (2020). Physical Plausibility of 6D Pose Estimates in Scenes of Static Rigid Objects. *European Conference on Computer Vision Workshops (ECCVW), 648-662.* DOI: 10.1007/978-3-030-66096-3_43
- **Bauer, D.**, Patten, T., & Vincze, M. (2020). Scene Explanation through Verification of Stable Object Poses. *ICRA 2020 Workshop on Perception, Action, Learning: From Metric-Semantic Scene Understanding to High-level Task Execution.*
- **Bauer, D.**, Patten, T., & Vincze, M. (2020). VeREFINE: Integrating Object Pose Verification with Iterative Physics-guided Refinement. *IEEE Robotics and Automation Letters (RA-L), 5(3), 4289-4296. With oral presentation at IROS 2020.* DOI: 10.1109/LRA.2020.2996059
- **Bauer, D.**, Patten, T., & Vincze, M. (2019). Monte Carlo Tree Search on Directed Acyclic Graphs for Object Pose Verification. *International Conference on Computer Vision Systems (ICVS), 386-396.* DOI: 10.1007/978-3-030-34995-0_35
- **Bauer, D.**, Patten, T., & Vincze, M. (2019). 6D Object Pose Verification via Confidence-based Monte Carlo Tree Search and Constrained Physics Simulation. *OAGM & ARW Joint Workshop, 153-158.* DOI: 10.3217/978-3-85125-663-5-31

*Equal contribution.

Chapter 2

Background

This chapter discusses related work in the areas of object pose estimation, refinement and verification in Section 2.1. We provide an overview of the datasets used for the experiments in this work in Section 2.2 and discuss the associated evaluation metrics in Section 2.3.

2.1 Related Work

A special emphasis is put upon methods that incorporate physical and scene-level considerations. We identify approaches that employ RL concepts similar to what we use in our proposed methods in the pose estimation context. In addition to work related to the pose estimation steps themselves, we discuss previous work in point cloud registration as it closely relates to pose refinement. Finally, we discuss work on explainability in the context of HRI.

2.1.1 Object Pose Estimation

A thorough discussion of the field is provided by Hodaň et al. in the course of the BOP Challenges [30]–[32]. This includes several datasets in a common format, metrics and tools for evaluation as well as a comparison of state-of-the-art methods that were submitted to the respective challenge iterations. We show a brief overview of these classical and learning-based pose estimation approaches.

Template-based and Feature-based Pose Estimation: Template-based methods are shown to perform well with texture-less objects [33], [34]. Pre-computed views of a 3D model of an object are matched to the observation at runtime. Conversely, these methods are more sensitive to novel view points not included in the templates. This problem is alleviated by the use of local feature descriptors. The correspondences between the features computed for the 3D model and the observation allow to determine the object's pose. In absence of texture, edge features may be used [35], [36]. Point Pair Features (PPF) have been shown to be well suited for the pose estimation task and the corresponding methods [10], [11], [37] achieve state-of-the-art performance when depth data is available.

Learning-based Pose Estimation: Building on the idea of feature-based pose estimators, learning-based methods are proposed that aim to learn suitable features [38], [39]. Encoder-decoder architectures are proposed that predict 3D locations on the corresponding model [9], [40]. Alternatively the location of keypoints outside of the object itself, such as 3D bounding

box corners, may be estimated [41]–[43]. To prune correspondences that relate to close-by objects in clutter, consideration of segmentation [41] or self-occlusion [44] is proposed to allow more robust matching and hence pose estimation. Related to template-based approaches, Sundermeyer et al. [45] leverage an autoencoder to learn a codebook of latent codes that correspond to specific observed object rotations. The translation [46] or full pose of the object may also be directly regressed [47], [48]. PoseCNN [8] jointly regresses object translation, rotation and segmentation mask from RGB images. The added depth information of RGB-D images is leveraged by combining appearance and geometric features for a joint regression in [12], [49]. Shi et al. [50] propose to restrict the space of admissible poses by considering the geometric stability (e.g., co-planarity) for patch-wise pose prediction. Additional contextual features capture the relation between patches. An important challenge for learning-based methods are symmetries, leading to ambiguous training targets. Several loss functions are proposed that are invariant to symmetry [9], [12], [22] or explicitly consider symmetry classes [50]. Manhardt et al. [51] predict a pose for each ambiguous case under (observed) symmetry, resulting in a pose distribution that collapses to a single estimate whenever the correct pose may be unambigously identified in the observation. The predicted distribution hence captures the method's uncertainty and allows to detect and explain ambiguous observations.

2.1.2 Point Cloud Registration and Object Pose Refinement

Previous work on object pose refinement exploits object segmentation, color or depth images as input modalities. These methods use hand-crafted or learned features, use a closed-form solution or learn an estimator to determine the relative pose. Moreover, methods for (depth-based) pose refinement commonly apply point cloud registration approaches. Closely related to our methods, the use of RL and scene-level considerations are proposed.

Point Cloud Registration: Besl et al. [17] propose a seminal approach in point cloud registration, the Iterative Closest Point (ICP) algorithm. First, the (closest) points in the source and target points clouds are matched. Then, a transformation that minimizes the error between the matched points is found in closed form. These steps are repeated until convergence. Many variants are proposed, e.g., considering surface normals [18], color [52] or using non-linear optimization [53]. To increase robustness to outliers, Gold et al. [54] propose to use *soft* correspondences that become increasingly strict (i.e., binary) in an annealing scheme. Kehl et al. [16] combine the ICP objective with a contour-matching objective based on the RGB observation. For a detailed overview of ICP-based methods, see [18], [55]. ICP may, however, only find local optima. A branch-and-bound variant [56] trades global optimality for increased runtime. Other global approaches use local features [57] in a RANSAC scheme [58] or directly optimize a global objective [59]. TEASER [60] accounts for outliers by a truncated least squares cost and allows to certify global optimality of the estimated registration.

Learning-based Point Cloud Registration: Based on the idea of ICP and its global variants, learning-based methods predict local features to determine a matching between the input clouds. Using this matching, the transformation is found either in closed form using differentiable Weighted SVD [61]–[64] or by optimization using stochastic gradient descent [65]. This enables the definition of end-to-end learnable registration pipelines. Notably, the method by Yew and Lee [63] additionally uses surface normals to compute PPF as input. While there is effort to extract more robust features [63], [65], these methods

typically use secondary networks that predict the sharpness of the match matrix to deal with imperfect correspondences and outliers [62], [63]. In contrast, another class of methods uses global features that represent whole point clouds and as such are more robust to imperfect correspondences. Seminal work in this direction by Aoki et al. [66] poses iterative registration as the alignment of global features using an interpretation of the Lucas-Kanade algorithm. A deterministic formulation that replaces the approximate with an exact Jacobian is proposed in [67], which increases the stability of the approach. The method in [68] allows one-shot registration of global features. Yuan et al. [69] learn Gaussian Mixture Model parameters for probabilistic registration.

Learning-based Pose Refinement: For object pose refinement, Wang et al. [12] combine learned point cloud and image-based features to regress a pose update. For each intermediary estimate, the PointNet-based features are updated to guide the next step. In contrast, the goal of pose refinement may also be viewed as aligning a rendering of the object under estimated pose to an observed image patch [14], [70], [71]. Seminal work by Li et al. [14] uses optical flow features to predict the refinement transformation to more closely align the given image patches. Similarly, contour alignment between a rendered and observed object view may guide incremental prediction of refined poses [70]. The approaches of Shao et al. [15] and Busam et al. [72] consider this as an RL task and learn a policy that predicts discrete refinement actions. Based on 2D segmentation masks, these agents learn to predict discrete refinement actions. An alternative use of RL is proposed by Krull et al. [73], learning a policy that efficiently allocates refinement iterations to a pool of pose hypotheses.

Scene-level Pose Refinement: However, these approaches consider a single target object at a time. Especially in heavily cluttered scenes that may occur, for example, during robotic manipulation, consideration of the full scene of objects is shown to be beneficial [74]. Labbé et al. [22] propose a global refinement scheme, incorporating all scene objects in a joint pose optimization and incorporate multiple views of the scene. Sui et al. [74] refine object poses by optimizing a graph defined by the geometric relations between interacting objects' contact surfaces. By fusing multiple views into a voxel grid, Wada et al. [75] consider the collision between scene objects in their refinement method. Alternatively, physics simulation may be used to test the plausibility of scene objects' poses [23], [76]. Desingh et al. [76] combine a particle filter formulation with rendering-based scoring of pose hypotheses to generate pose estimates for scenes of box-like shapes. Mitash et al. [23] leverage the simulated dynamics of the scene objects to improve pose hypotheses and verify the best aligned combinations with respect to the observed depth image.

2.1.3 Hypotheses Verification in Pose Estimation

Such pose hypotheses verification is especially needed in a robotics context, where false-positive detections and inaccurate poses may result in faulty robotic interactions. However, determining erroneous hypotheses or picking the best hypothesis among the top-n estimates requires a sort of scoring of the pose hypotheses.

Point Cloud-based Pose Verification: Drost et al. [10] use a clustering-based verification stage to refine pose estimates. In [11], a pool of 200 object pose hypotheses is generated using a PPF-based pipeline. Each hypothesis is then refined using Projective ICP and a two-step verification to determine the best estimate. Wang et al. [12] jointly predict a pose confidence

score with per-pixel object pose estimates. The highest scoring estimate is selected and refined. Sui et al. [74] compute a geometric consistency score with respect to the estimated scene, allowing to prune false positives and merge overlapping pose candidates. Aldoma et al. [20] use a Simulated Annealing scheme to select a subset of hypotheses that fits the observation, using geometrical cues together with cues for clutter and conflicting correspondences with respect to the observation. In follow-up work [21], different meta-heuristics are compared and the cost function is expanded to consider color cues.

Image-based Pose Verification: Illustrating the use of the analysis-by-synthesis paradigm, the evaluation of candidate solutions in terms of depth-based alignment may be performed by exhaustively searching through all possible object pose hypotheses combinations in a brute-force manner [19]. More efficiently, Mitash et al. [23] combine such rendering-based scoring with Monte Carlo Tree Search (MCTS) to search through the pool of pose hypotheses. Given a rendered view of the object under a hypothesized pose and the observation, Krull et al. [73] train a CNN to predict two different hypotheses scores, used to guide refinement and for the selection of the final estimate.

Simulation-base Pose Verification: Mitash et al. [23] integrate physics simulation in the evaluation of candidate solutions. While this potentially improves the solution's plausibility and highlights inconsistencies, it dramatically slows down the run time to a reported 30s per frame. Following a similar approach, Sallami et al. [77] run a simulation for a few steps and evaluate the displacement from the observed scene. Thereby, pose hypotheses are improved while a displacement-based stability check allows to reject erroneous hypotheses.

2.1.4 Explainability and Explanation Strategies

A robot that is able to provide explanations for its understanding of a scene and its resulting actions may allow users insight into its inner workings. Especially when the robot's behavior seems unexpected to users, such as when commands are misunderstood or actions fail, explanations may help to reduce the users' perceived risk – and in turn increase the trustworthiness that users ascribe to the robot.

Explainability and Perceived Trustworthiness: Andras et al. [78] argue that this happens not only in unexpected cases but as well in the initial phase of the interaction, allowing users to establish a mental model of the robots' inner workings. Several studies show that explanations, thereby, may support the trustworthiness of artificial agents [2] and robots in particiular [1], [3]. Users' biases towards the robot and its functioning as well as misleading explanations might, on the contrary, reduce the perceived trustworthiness [79]–[81].

Multi-modality and Interactivity: Textural explanations based on natural language, followed by visualizations, represent the most commonly used modalities in human-agent interactions [82]. Combining different modalities into a consistent multi-modal explanation strategy may increase users' understanding of these explanations, as shown for example for image classification tasks [83], [84]. Lakkaraju et al. [79], [85] propose to leverage interactivity as an alternative strategy that may increase the effectiveness of explanations. Dune et al. [81] argue that the resulting dialogues between, for example, the robot and the user allow the latter to examine the reasoning provided in the explanation. Such an examination may also be used to correct mistakes based on the users' feedback [86]. Moreover, a combination of multi-modal and interactive explanation strategies is proposed for human-agent settings [87].

2.2 Datasets

In this work, we approach the task of object pose estimation and the related problem of point cloud registration. As in related work in these areas, we use the following datasets for evaluation, illustrated in Figures 2.1, 2.2 and 2.3.

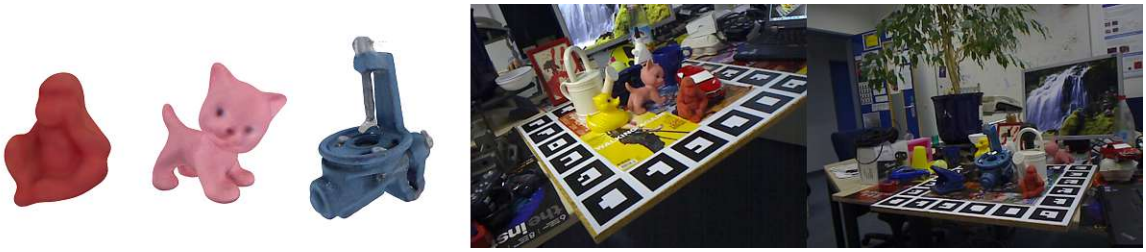


Figure 2.1: Examples from the LINEMOD dataset [34]. For OCCLUDED [48], the remaining objects in the *benchvise* scene (right) are annotated.

Figure 2.2: Examples from the YCB-Video dataset [8].

Figure 2.3: Examples from the Rutgers Extended RGBD dataset [23].

2.2.1 Object Pose Estimation Datasets

Common pose estimation datasets consist of several 3D models, potentially with annotated symmetry transformations, ground-truth segmentation masks and object poses for each frame. Per frame, one or multiple objects may be observed in isolation or in clutter.

LINEMOD and OCCLUDED: The LINEMOD dataset [34] is used to evaluate the single-object setting. It consists of 15 test scenes showing one object in a cluttered environment each but with only minor occlusion of the target object. For OCCLUDED [48], the remaining objects in the *benchvise* scene from LINEMOD, featuring significant occlusion, are annotated.

YCB-Video: The YCB-Video dataset (YCBV) [8] is used for the multi-object setting. The dataset exhibits clutter as well as isolated, 2- and 3-object support relationships. YCB-Video contains 92 scenes. The 12 test scenes consist of 3 to 6 objects from the YCB object set [88].

Rutgers Extended RGBD: In addition, the Rutgers Extended RGBD dataset [23] is used for the multi-object setting in Chapter 5. The dataset exhibits clutter as well as isolated, 2- and 3-object support relationships. It uses Amazon Picking Challenge objects and features three objects per scene.

2.2.2 Point Cloud Registration Datasets

In the context of depth-based refinement, we also consider the related task of point cloud registration in Chapter 6.

ModelNet40: As in prior work in point cloud registration, we evaluate on ModelNet40 [89] featuring synthetic CAD models from 40 classes. We follow related work [61], [62], [66] and use the point clouds generated by Qi et al. [90] by sampling the surface of the CAD models.

ScanObjectNN: To additionally evaluate generalization from synthetic to real data, we consider the ScanObjectNN dataset [91], featuring observations captured from an RGB-D sensor. We use the point clouds from the segmented-objects split of the ScanObjectNN dataset.

Figure 2.4: Example point clouds from the ModelNet40 dataset [89], [90].

Figure 2.5: Example point clouds from the ScanObjectNN dataset. Adapted from [91].

2.2.3 Robotic Grasping Dataset

The YCB objects [88] provide a standardized, readily available and thus commonly used set of 3D models and real objects to evaluate robotic grasping. Using the subset of the objects used in the YCB-Video dataset, we moreover have access to large amounts of training data and may evaluate the proposed methods on the YCB-Video dataset too. Reproducible experimental conditions are ensured by using the GRASPA scene layouts [92] that indicate where to place objects with respect to one another.

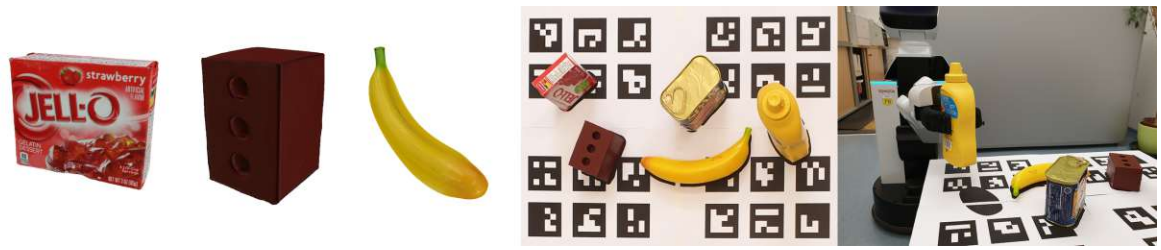


Figure 2.6: Example of a GRASPA layout [92] using YCB objects [88].

2.3 Task Definition and Metrics

The metrics in our experiments are commonly used in related work [32], [34], [61], [63]. We provide their definition in condensed form in the following section.

First, we formally define the pose estimation and refinement tasks and introduce some notation. Assume an object model $M = \{m_i \in \mathbb{R}^3\}$ and an observation of the object $O = \{o_i \in \mathbb{R}^3\}$ to be given. The model M is offset from the observation O by an unknown rigid transformation $T = [R, t]$, where R is a rotation from the special orthogonal group $SO(3)$ and t is a translation vector in $\mathbb{R}^3$. The task of object pose estimation is then to determine this transformation $\hat{T}$ such that

$$O = \hat{T} \otimes M, \tag{2.1}$$

that is, such that the model M aligns with the observation O. The operator $\otimes$ indicates the application of the transformation by, for example, homogeneous coordinates, rotation matrix or quaternion representation. In this simplest case both M and O are identical but, in general, one or both point sets are noise afflicted, their order may not correspond, not every point in the model may have a correspondence in the observation and the number of points may not be identical.

In *iterative registration* or *iterative refinement*, n steps are taken to compute this transformation. In every step i, a rigid transformation $\hat{T}_i$ is estimated and applied by

$$M_i = \hat{T}_i \otimes M_{i-1}. \tag{2.2}$$

The goal is that the final estimate after n steps is again

$$M_n = \hat{T}_n \otimes ... \otimes \hat{T}_1 \otimes M = \hat{T} \otimes M = O. \tag{2.3}$$

2.3.1 Transformation-based Metrics

Shotton et al. [93] propose to compute the translation and rotation errors of pose estimates individually and to apply a threshold, for example by 5cm and 5deg, to determine whether an estimate is to be considered correct. Similary, in the related field of point cloud registration [61], [66], the mean absolute and the isotropic transformation errors are used.

Mean Absolute Error (MAE): The MAE between a vector of estimated $\hat{v}$ and true values v is defined as

$$MAE_v = \text{avg}\,|\hat{v} - v|, \tag{2.4}$$

where v is either the vector of Euler angles in degrees representing the rotation or the translation vector.

Isotropic Error (ISO): While MAE considers axes individually, ISO is computed over the full rotation and full translation. For the rotation error ISO_R, we compute the angle of the residual rotation matrix by

$$ISO_R = \arccos \frac{trace(\hat{R}^{-1}, R) - 1}{2} \tag{2.5}$$

and Euclidean distance between the estimated and true translation gives

$$ISO_t = ||\hat{t} - t||_2. \tag{2.6}$$

2.3.2 Model-based Metrics

Object pose estimation and evaluation thereof are commonly based on the alignment of the 3D object model [32]. This remedies the issue of non-identical point sets in Equation (2.1) by representing both by the known object model M, transformed by the ground-truth pose T and the estimated pose $\hat{T}$, respectively. This allows to directly compare the two point sets we will denote as $X = T \otimes M$ and $Y = \hat{T} \otimes M$.

Hausdorff Distance: The Hausdorff distance of two point sets X,Y is defined as

$$\begin{aligned} d_H(X,Y) &= \max(d_h(X,Y), d_h(Y,X)), \text{ where} \\ d_h(X,Y) &= \max_{x \in X} \min_{y \in Y} ||x - y||_2. \end{aligned} \tag{2.7}$$

Intuitively, d_H is an upper bound for the displacement between the two; the largest distance between the closest points.

Chamfer Distance ($\tilde{CD}$): The Chamfer distance, used for evaluation in point cloud registration [63], is defined as

$$CD(X,Y) = \operatorname*{avg}_{x \in X} \min_{y \in Y} ||x - y||_2^2. \tag{2.8}$$

Modified Chamfer Distance $\tilde{CD}$: A modified variant is proposed in [63], where additional noise is applied to yield $\tilde{X}$ and $\tilde{Y}$. Based on the Chamfer distance, it is defined as

$$\tilde{CD}(X,Y) = CD(\tilde{X}, Y) + CD(\tilde{Y}, X). \tag{2.9}$$

Average Distance of Model Points (ADD) and Average Distance of Model Points with Indistinguishable Views (ADI): The most used metric in related work on pose estimation, the ADD, is proposed in [34]. It is defined as the mean distance between corresponding points

$$ADD = \operatorname*{avg}_{x \in X, y \in Y} ||x - y||_2. \tag{2.10}$$

In addition, Hinterstoisser et al. [34] propose to account for symmetrical true poses by

$$ADI = \operatorname*{avg}_{x \in X} \min_{y \in Y} ||x - y||_2. \tag{2.11}$$

The ADI recall for a specific precision threshold and N test samples is

$$ADI_{th} = \frac{1}{N} \sum_i \begin{cases} 0, & ADI_i > th \\ 1, & ADI_i \leq th, \end{cases} \tag{2.12}$$

where ADI_i is the ADI of the i^{th} test sample. The ADD recall is computed analogously. Note that, as Chamfer distance, the ADI metric implicitly considers symmetry by considering the closest point pairs.

Maximum Symmetry-Aware Surface Distance (MSSD): If the symmetry transformations for a given object are known, the MSSD [94] allows to explicitly evaluate the error with respect to any equivalent, symmetrical pose of the model by

$$MSSD = \min_{S \in \mathcal{S}} \max_{s \in S} ||s - y||_2, \tag{2.13}$$

where $S = T \otimes T_s \otimes M$ and $\mathcal{S}$ is the set of all symmetric versions of the model for the given symmetry transforms T_s. Note that the definition of T_s may consider geometrical symmetries or, more strictly, only textural symmetries. The latter are not observable by depth-based methods.

The following projection-based metrics consider the perceptual error with respect to the observation, i.e., they account for camera viewpoint and observablility in the image.

2D Projection (Proj2D): For this metric, we project X,Y onto the 2D observation for which the pose is estimated using the camera's intrinsic matrix K and compute the 2D projection error [48] as

$$Proj2D = \operatorname*{avg}_{x \in X, y \in Y} ||proj(x) - proj(y)||_2, \tag{2.14}$$

where $proj(p) = K \otimes p$ is the projection of a point p to the 2D image using K.

Maximum Symmetry-Aware Projection Distance (MSPD): Analogous to MSSD, we may also account for the symmetry transformations when computing the projection distance [32] by

$$MSPD = \min_{S \in \mathcal{S}} \max_{s \in S} ||proj(s) - proj(y)||_2, \tag{2.15}$$

where again $S = T \otimes T_s \otimes M$ and $proj$ is defined as for Proj2D.

Visible Surface Discrepancy (VSD): In contrast, the Visible Surface Discrepancy [30], [95] (VSD), considers the discrepancy of the rendered depth images of the object under ground-truth pose $\hat{I}_d(T)$ and estimated pose $\hat{I}_d(\hat{T})$ by

$$VSD = \text{avg}_{p \in V(\hat{T}) \cup V(T)} \begin{cases} 0, & \text{if } p \in V(\hat{T}) \cap V(T) \text{ and } \Delta(p) < \tau, \\ 1, & \text{otherwise.} \end{cases} \tag{2.16}$$

The visibility V under a given pose is computed with respect to the observed depth image I_d. The absolute difference $\Delta(p)$ between the rendered images at a pixel p is bound by a misalignment threshold τ.

2.3.3 Summary Metrics

The presented metrics may be evaluated at different correctness thresholds. To summarize the performance over a wide range of thresholds, the following summary metrics are proposed.

Area under the Recall Curve (AUC): The average precision, defined in PASCAL VOC [96] as Area Under Curve (AUC), evaluates a metric at uniformly spaced precision

thresholds up to a maximum threshold. This results in a monotonically increasing precision-recall curve. For example, ADI AUC is then defined as the area under this curve by

$$AUC = \frac{1}{th_{max}} \sum_{th \in [0:\Delta:th_{max}]} ADI_{th} \cdot \Delta, \tag{2.17}$$

where Δ is the threshold spacing. A maximal threshold $th_{max} = 0.1d$ is commonly used with the ADD and ADI metrics [8], where d is the diameter of the model and computed as maximal distance between any two points [34]

$$d = \max_{x_1, x_2 \in X} ||x_1 - x_2||_2. \tag{2.18}$$

Average Recall (AR): In addition to considering multiple thresholds, this metric uses three different error functions, namely, the MSPD, the MSSD and the VSD. Per error function, the average recall rates over 10 thresholds are computed. The overall performance score (AR) is the average recall over the three resulting sub-scores. See [32] for a definition of the thresholds used per sub-score.

2.3.4 Grasping Metrics

The evaluation of grasp poses and grasp quality may be carried out extensively using a battery of metrics [92], [97]. Instead, we evaluate the suitability of pose estimates for the downstream tasks in the grasping pipeline outlined in Figure 1.2. Similar to Wang et al. [12], we resort to the evaluation of whether the grasp planning and the grasp execution succeed. In addition, we leverage the layouts proposed by Bottarel et al. [92] to ensure repeatability between experimental runs using different methods and reproducibility of the evaluation. Manually annotated grasp poses per target object are transformed to the scene using the estimated object poses. Grasp poses for which the gripper would collide with the observed octomap are pruned. Trajectories for all collision-free grasp poses are planned using MoveIt [98]. If at least one plan is found, this is counted as a *found* grasp. A grasp is considered a *success*ful grasp if the plan can be executed, i.e., the object is grasped and remains stable in the robot's gripper. The *found* and *success* rates are averaged over 10 grasp attempts per object – 5 grasps are attempted for a given pose and an additional 5 for a symmetric pose of the object.

Chapter 3

Definition of Visual and Physical Plausibility of Object Poses

Object pose estimation approaches are evaluated by how accurately the surface of a target object under estimated and true pose aligns. This visual plausibility alone, however, cannot disambiguate equally misaligned but physically implausible object poses. For example, an estimate of an object resting on a table and an object floating in thin air may both exhibit the same visual misalignment, however, only one is physically plausible. Two basic assumptions on physical plausibility are that all objects in a static scene are non-intersecting and (indirectly) in contact with a static support surface. More generally, the static equilibrium of a scene is considered a condition for physical plausibility. But also physical plausibility by itself is inherently ambiguous. For example, there is a vast number of plausible poses for an object to rest on a plane.

Our goal is therefore to jointly consider visual and physical plausibility to overcome the pose ambiguity that occurs when using either aspect in isolation. Under ambiguous surface alignment, the physically more plausible pose should be preferred – under ambiguous physical plausibility, the closer surface alignment should be preferred. We propose definitions of both aspects in this chapter, discuss how they are computationally evaluated and leverage them throughout this work at different steps of the pose estimation pipeline. The information gathered to determine an estimate's plausibility may moreover be used to explain a robot's understanding to users.

Contributions

- Definition of visual plausibility as rendering-based alignment using depth and normal information
- Definition of physical plausibility using the convex hull of an object for planar support geometry
- Definition of physical plausiblity using the contact points of an object for general support geometry

The content of this chapter is based on previously published work in [24]–[27].

3.1 Visual Plausibility

Object pose estimation and evaluation thereof are commonly based on the alignment of a 3D object model [32]. The most used metric in related work, the Average Distance of Model Points (ADD) [34], measures the mean distance between corresponding model points under estimated pose $\hat{T}$ and under ground-truth pose T. In contrast, the Visible Surface Discrepancy [95] (VSD), considers the discrepancy of the rendered depth images of the object under estimated pose $\hat{I}_d(\hat{T})$ and under ground-truth pose $\hat{I}_d(T)$.

Building on this idea, we assume that the observed and rendered view of a scene should align for it to be visually plausible. We define a visual-alignment score $\bar{a}$ in [24] that quantifies the average alignment between the object in the observed (i.e., under ground-truth pose T) and rendered depth and normal images under the estimated pose $\hat{T}$. As illustrated in Figure 3.1, $\bar{a}$ is computed over all pixels with valid depth values, defined as $V = I_d > 0 \cup \hat{I}_d(\hat{T}) > 0$, by

$$\bar{a} = \frac{1}{2}\left(\text{avg}_{p \in V} a_d(p) + \text{avg}_{p \in V} a_n(p)\right), \tag{3.1}$$

with depth-based alignment a_d and normal-based alignment a_n per pixel p defined as

$$a_d(p) = \begin{cases} 1 - \frac{|d - \hat{d}|}{\tau}, & \text{if } |d - \hat{d}| < \tau \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

$$a_n(p) = \begin{cases} 1 - \frac{1 - n \cdot \hat{n}}{\alpha}, & \text{if } 1 - n \cdot \hat{n} < \alpha \\ 0, & \text{otherwise,} \end{cases} \tag{3.3}$$

where $d \in I_d$ is the depth value and $n \in I_n$ is the corresponding normal at pixel p in the observation. The corresponding values in the rendered image are denoted $\hat{d} \in \hat{I}_d(\hat{T})$ and $\hat{n} \in \hat{I}_n(\hat{T})$. The parameters τ and α limit the maximal admissible discrepancy.

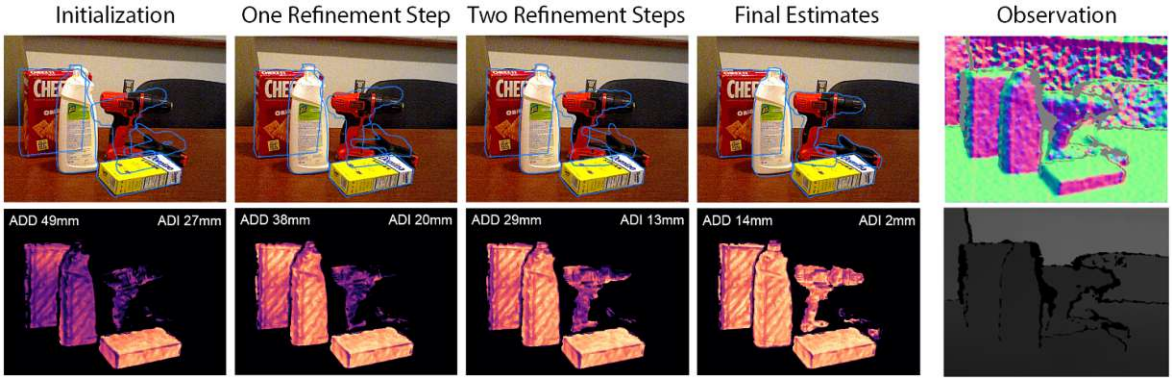


Figure 3.1: Visualization of the visual-alignment score $\bar{a}$ for multiple steps of the refinement approach in [25]. The observed normal I_n and depth image I_d are shown in the last column. In the visualizations in the bottom row, a more yellow color indicates a higher alignment score between estimate and observation. ADD/ADI are reported as the mean over per-object distances. [25].

We implement an computationally efficient evaluation of Equation (3.1) using OpenGL. First, the depth image $\hat{I}_d$ and the normal image $\hat{I}_n$ of the object(s) under the estimated pose

$\hat{T}$ are rendered to a texture. Second, a_d and a_n are computed per pixel and again stored in a texture. By generating the mipmap levels of this per-pixel score texture, the sums for $|V|$, a_d and a_n are computed. Reading from a high mipmap level, only the sums need to be transferred from the GPU to the CPU for final averaging instead of slowly reading-back the full depth and normal images. Thereby, we are able to drastically reduce the evaluation time from 7-9ms (using full images) to 1-2ms on a NVIDIA GTX 1080Ti for images of 640x480 pixels.

Figure 3.2 illustrates the effect of different pose errors on the visual-alignment score.

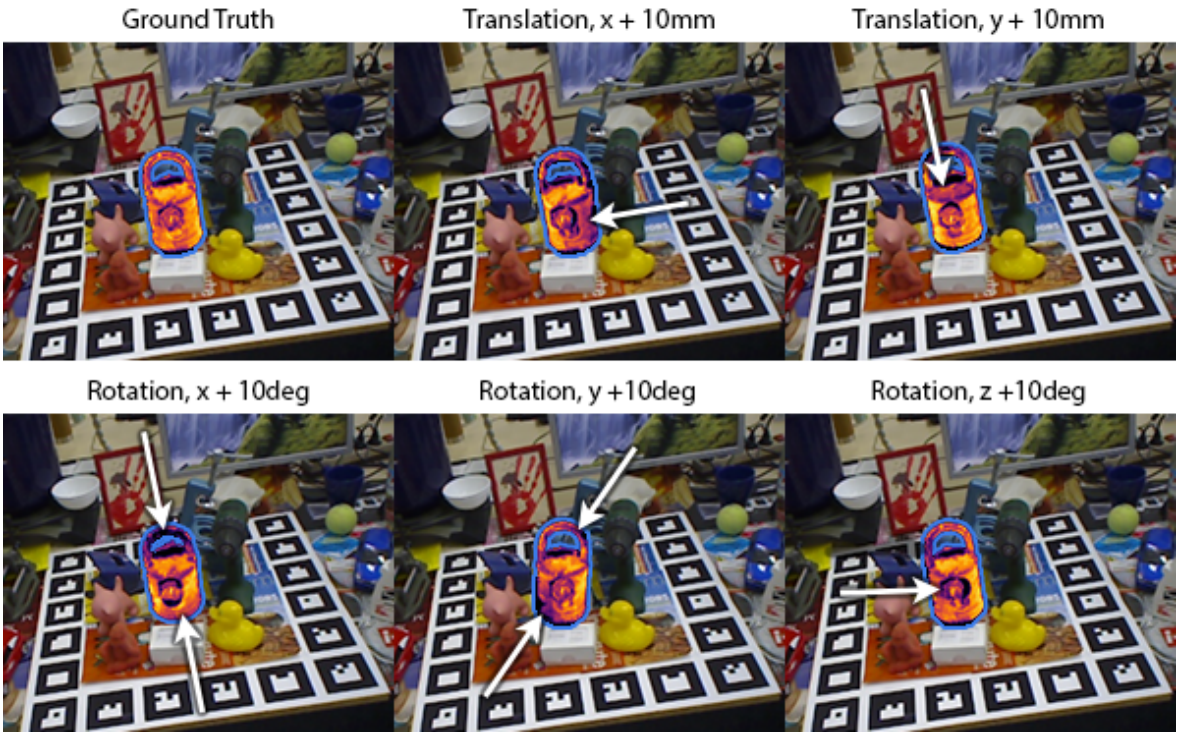


Figure 3.2: Visualization of the effect of a varying pose error on the visual-alignment score $\bar{a}$. The arrows indicate areas where the per-pixel score is lowered as a result of the pose error. The darker the color, the worse the alignment – black areas are misaligned by more than the threshold.

3.2 Physical Plausibility

Estimates of floating or intersecting objects may yield the same visual-alignment score as physically plausible estimates of the same error magnitude. In robotic grasping, implausible pose estimates that, e.g., intersect with the scene may result in a collision of the robot with its environment. We also conjecture that, for human observers, pose estimates that follow physical principles are more intelligible. We thus argue that physically plausible pose estimates should be preferred under visual ambiguity.

In the following, we assume that the scene objects are known, rigid (they do not deform under stress) and static (they do not move with respect to a known frame of reference). We furthermore assume that the only external force acting on the scene is gravity.

3.2.1 Planar Support

We observe that stable object poses in static scenes are only a small subset of the full SE(3) group. Rather, there are a number of stable clusters that are determined by the geometry of both the environment and the object itself. For example, a cube can only stably rest on a supporting plane on one of its 6 faces. This drastically reduces the space of physically plausible poses: (1) The translation must be such that the cube is in contact with the supporting plane and (2) the rotation must be such that one of the cube faces is co-planar to the supporting plane. This constrains the admissible object poses of the cube to only 6 discrete possibilities – up to in-plane translation and in-plane rotation.

3.2.1.1 Geometric Constraints

More generally, assume an object O is placed on an infinitely large plane, illustrated in Figure 3.3 (left). The object can rest on any point on its boundary $\mathbf{bd}O$. For 3D meshes, this boundary is piece-wise continuous, as $\tilde{O}$ in Figure 3.3 (mid). Following the definitions in [99], we can define the set of rest poses $\mathcal{H}$ of the object as a convex polytope given by the intersection of all supporting hyperplanes

$$H(x_0) = \{x | a^T x = a^T x_0\} \tag{3.4}$$

where $x_0 \in \mathbf{bd}O$, $a \neq 0$ and a satisfies

$$a^T x \leq a^T x_0, \forall x \in C. \tag{3.5}$$

Each hyperplane yields one potential rest pose hypothesis, with the object's rotation given by a and its height above the plane by x_0. The intersection of the hyperplanes is equivalent to the convex hull $\tilde{C}$ of the object. It follows that to yield all rest pose estimates, an object model can be of arbitrary precision as long as the convex hull remains the same.

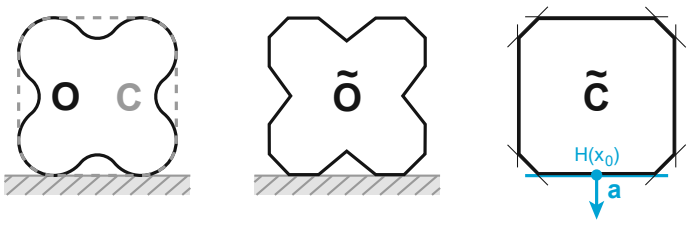


Figure 3.3: Geometric constraints for planar support. Left: A non-convex object O and an equivalent convex object C. Mid: The mesh $\tilde{O}$ of O. Right: The convex hull $\tilde{C}$ of $\tilde{O}$ with one of the support hyperplanes $H(x_o)$ (blue). [27].

For a non-convex object O, any tangent hyperplane on the non-convex subset of the boundary will intersect O and thus violate the inequality in Equation (3.5). See the hyperplane in Figure 3.4 (mid) for an example. Under the assumption of an infinitely large plane, however, only the convex subset of its boundary need to be considered for computing $\mathcal{H}$. Therefore, the concave parts can be of arbitrarily low precision and we approximate O by its convex hull.

The geometric considerations also apply to finite planes if all faces of the convex hull lie within the plane's boundary. This assumption holds for many household scenarios, for example, with objects placed on shelves, tables or in stacks of box-like objects.

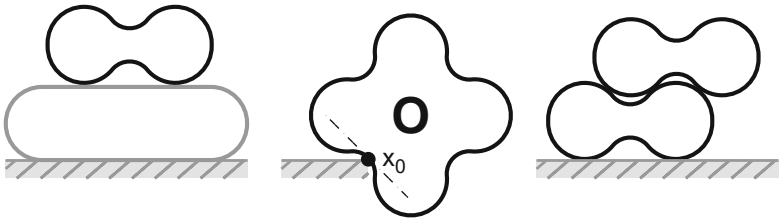


Figure 3.4: Geometric constraints for non-planar support. Left: The assumption holds also for locally-planar object interactions. Mid: If the plane assumption is violated, O may also rest on non-convex regions. Right: Complex object interactions need to be considered on-the-fly or as in Section 3.2.2. [27].

3.2.1.2 Determining Stable Poses

While $\mathcal{H}$ allows the potential rest poses to be determined based on the object's geometry, stable rest poses also depend on the object's physical properties – most importantly its center of mass (CoM), $c_m \in \mathbb{R}^3$. In general, the CoM will not be known. We assume the objects' to be of uniform density and compute the CoM as the centroid of all points in an object's volume.

In addition, due to the piece-wise continuous boundary, a single stable pose of the precise model can give rise to multiple stable poses of the approximate model. For example, assuming a centered CoM, object O in Figure 3.3 (mid) could balance on exactly one point per corner. Its mesh $\tilde{O}$, however, stably rests on any point of the face that approximates a corner. Considering each stable pose of the approximate model is therefore intangible. Rather, as discussed in [100], similar stable poses can be reduced to one representative.

When a sufficiently large support plane can be assumed, the stable poses only depend on the object's geometry and CoM. Therefore, the stable poses only need to be generated once. If this assumption is violated, as illustrated in Figure 3.3 (mid and right), the object can however rest on any of its boundary points.

3.2.2 General Support

We consider the contact points for general supporting symmetry and concave objects. In these cases, additionally to rigidity and static scenes, we further assume that friction-afflicted surface contacts are the only type of support that transmits the external force $f_{ext} = mg$.

3.2.2.1 Feasibility Constraints

We describe the interaction of a target object with the scene by how far points on its surface are from the surface of other objects. Let this surface distance be

$$d(x_i, S) = \min_{y_i \in \mathbf{bd}S} \|x_i - y_i\|_2, \tag{3.6}$$

where $x_i \in \mathbf{bd}O$ is a point on the surface of a target object O and d is its Euclidean distance to the closest point on the surface of scene object S.

Based on the surface distance d, we define the set of contact points of O as

$$\mathcal{C}(O) = \bigcup_{S \in \mathfrak{S}} \{x_i \in \mathbf{bd}O | d(x_i, S) < \varepsilon\} \tag{3.7}$$

and the set of intersecting points of O as

$$\mathcal{I}(O) = \bigcup_{S \in \mathfrak{S}} \{x_i \in (\mathbf{bd}O \cap \mathbf{int}S) | d(x_i, S) > \varepsilon\}, \tag{3.8}$$

where $\mathfrak{S}$ is the set of scene objects (excluding target O) and ε is a small constant to account for the finite accuracy of the approximation of the surfaces of O and $S \in \mathfrak{S}$, e.g., through a mesh or point cloud. $\mathbf{int}$ denotes the interior of an object.

We define a pose as *feasible*, if the object is in contact with at least one other object (not *floating*) and does not intersect any other object (not *intersecting*). Thus, for O to be under a feasible pose, the conditions

$$\text{not floating:} \quad |\mathcal{C}(O)| > 0, \tag{3.9}$$

$$\text{not intersecting:} \quad |\mathcal{I}(O)| = 0 \tag{3.10}$$

must be satisfied.

3.2.2.2 Stability Constraints

We define the pose of a target object as *stable*, if it is in static equilibrium (SE) when supported at its contact points $\mathcal{C}(O)$. Let m be the the mass of the object and $c_m \in \mathbb{R}^3$ its center of mass. Then, in the described scenario, the conditions for an object to be in SE are [101], [102]

$$\text{force balance:} \quad \sum_i f_i + f_{ext} = \sum_i f_i + mg = 0, \tag{3.11}$$

$$\text{torque balance:} \quad \sum_i (c_m - x_i) \times f_i = 0, \tag{3.12}$$

$$\text{admissible contact force:} \quad f_i \in \mathcal{K}, \tag{3.13}$$

where $f_i \in \mathbb{R}^3$ is the contact force at $x_i \in \mathcal{C}(O)$ and $\mathcal{K}$ is a friction cone defined [101] as

$$\sqrt{(t_i^\top f_i)^2 + (b_i^\top f_i)^2} \leq \mu_i n_i^\top f_i, \tag{3.14}$$

using the static friction coefficient μ_i and the tangential plane at x_i spanned by $t_i, b_i, n_i \in \mathbb{R}^3$. Efficient algorithms that find a set of f_i such that Equations (3.11)–(3.13) are satisfied, i.e., to test for SE, are proposed in related work [101], [102].

The stability constraints may be approximated using the "support polygon principle" [103], where the *support polygon* is defined as the convex hull of the projection of the contact points $\mathcal{C}$ onto the supporting plane. If the projection of the center of mass lies within the support polygon, the object is considered to be in SE [103], [104].

Alternatively, in [25] we consider the support polygon with respect to the supported points

$$\mathcal{S}(O) = \{x_i \in \mathcal{C}(O) : \cos(n_y(x_i) \cdot g) < 0\}, \tag{3.15}$$

where $y(x_i)$ is the closest point to x_i in the scene and $n_y(x_i)$ is its surface normal. Thereby, only the contacts onto which a force may be exerted in gravity direction g are considered.

3.2.2.3 Physically Plausibility Constraints

Note that as the results of object pose estimation are, in general, not physically plausible, they may result in infeasible scene configurations. Such poses are not naturally covered by the notion of stability. Rather, feasibility is a necessary condition for stability. If both feasibility and stability are satisfied, we define this as a physically *plausible* pose. The used critical point sets and conditions are illustrated in Figure 3.5.

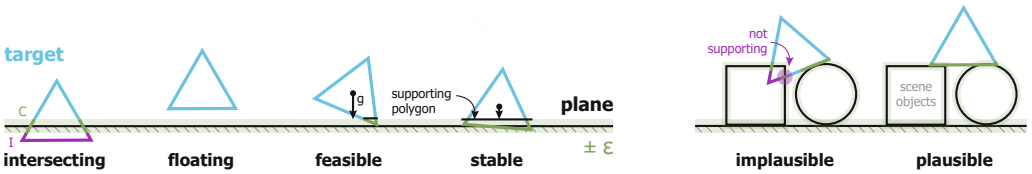


Figure 3.5: Definition of physical plausibility based on critical points for a single object (left) and a scene (right). If feasible, the CoM projected in gravity direction must intersect the support polygon (convex hull of supported points) to be considered stable. [26].

Using the presented considerations, we can binarily classify whether the respective conditions are satisfied by evaluating the contact and intersecting points of an object under a given pose. This may be used to verify pose estimates, e.g., before grasping a related object or to define a reward as in [25].

3.2.2.4 Determining Stability by Contact Points

For the latter application, we approximate $\mathbf{bd}O$ by sampled points on this surface. In addition, to simplify the evaluation of Equation (3.8), we consider the surface distance d to be negative for points in the interior of other scene objects.

We assume a plane as underlying static support, e.g., a floor, table or shelf that (indirectly) supports the scene objects. To determine the surface distance between scene objects, we transform all corresponding object point clouds to this plane. The surface distance with respect to the plane is now trivially obtained by the z-coordinate of the queried point x_i. For the distance between objects, the nearest neighbor in each scene object is computed per queried point. The point lies within the corresponding scene object if

$$\cos(n_y(x_i) \cdot (y - x_i)) > 0, \tag{3.16}$$

that is, if the normal of the nearest neighbor $n_y(x_i)$ points in the same direction as the vector from x_i to the nearest neighbor y. To make this test more robust, we compute it for the k nearest neighbors and consider the point to lie inside if it holds for a certain fraction of k. Note that a similar quorum check is used to compute the supported points $\mathcal{S}(O)$ in Equation (3.15). The final surface distance per point x_i is computed by taking the minimum over the distances to all scene objects (and the plane). The critical points of the queried point cloud are then computed using Equations (3.7), (3.8) and (3.15), with the gravity direction g given by the normal of the underlying supporting plane. The convex hull for applying the support polygon principle may be computed using the Quick Hull algorithm [105].

Highlights

- Efficient scoring of object pose hypotheses using rendering-based visual plausibility
- Determining the statically stable poses of an object mesh by analyzing its convex hull
- Determining physical plausibility using the interaction points of an object and its scene

Chapter 4

Object Pose Estimation through Verification of Stable Object Poses

Co-inhabiting spaces with humans, a robot will be presented with novel object instances on a daily basis. A user might try-out a new brand of cereal (intra-class variation) or bring home a new gadget (novel class). If a robot is to remain helpful under these conditions, it needs to be able to quickly reason about and interact with new object instances. While object detection approaches generalize to novel instances of a known class, object pose estimation methods typically deal with a few a-priori known instances. Handling novel instances or classes, thus, requires expensive data acquisition and training.

If we consider static scenes, we observe that most objects rest on a locally-planar surface and such objects only occur in a limited set of stable poses. The geometry of both the environment and the object as well as their physical interaction heavily restrict the space of admissible object poses, as discussed in Chapter 3. Based on this, we argue that approximate 3D models are sufficient to generate initial object pose estimates. We use a rendering-based hypotheses verification approach to determine the best estimate and, thus, the most plausible stable pose of an object. This allows new instances to be quickly added by requiring only a rough model of their geometry, in contrast to previous work that needs vast amounts of data to retrain networks or requires precise (textured) models.

Contributions

- Computing the set of stable poses using an approximate 3D model, physics simulation and clustering
- Demonstrate that a combination of visual and physical plausibility allows to limit the space of potential poses, simplifying the pose estimation problem
- Propose 3D model-based pose estimation using rendering-based verification to determine the stable pose with highest fit to an observed scene

The content of this chapter is based on previously published work in [27].

4.1 Pipeline: Pose Estimation as Verification of Stable Poses

Our premise is that the combination of physics simulation (to determine stable object poses) with hypotheses verification (to select the object pose that best fits the observation) is sufficient to compute accurate pose estimates. This replaces expensively trained object pose estimators, reducing the amount of data a robot requires to reason about and interact with a novel object instance from thousands of images to a single rough 3D model. The proposed pipeline consists of three steps: Detection and instance segmentation, computation of stable pose hypotheses (Sec. 4.2) and hypotheses verification for selection of the final pose estimate (Sec. 4.3). The object-based pose estimation pipeline illustrated in Figure 4.1 expects an RGB-D image and a set of approximate 3D meshes as input. In addition, we assume objects to rest on a locally-planar surface (ground or other object).

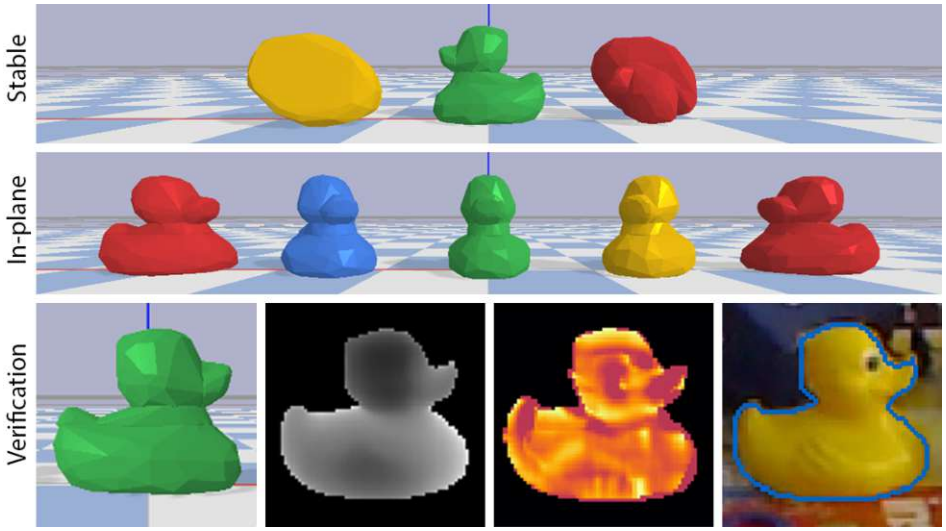


Figure 4.1: Object pose estimation by verification of stable poses. Top: Stable pose hypotheses. Mid: Rotation hypotheses for one stable pose. Bottom: Rendered depth of a hypothesis, its verification score and the selected object pose estimate. [27].

For use in a flexible scene explanation pipeline, the used detector needs to be able to generalize to novel instances of a set of known classes, i.e., it needs to deal with intra-class variation. A vast body of work tackles this problem and several off-the-shelf methods are available. For our experiments, we use pixel-wise instance segmentation and object labels precomputed using PoseCNN, as provided by Xiang et al. [8] and Li et al. [14].

The physics simulation of the detected objects requires a fixed supporting geometry. We consider table-top scenarios and assume objects are resting on a supporting plane. Therefore, in addition to the detection of the objects in the scene, we compute the supporting plane using plane segmentation. We generate a point cloud from the depth image, remove all points that belong to the object segments and use a MSAC scheme to fit a planar model into the remaining point cloud. The gravity vector is assumed to be normal to this plane. In larger scale scenarios, the actual supporting structures could be detected and fit to the observation, for example shelves or kitchen counters, or retrieved from a robot's object-based map.

4.2 Stable Pose Computation

We choose a *simulate-and-cluster* approach to determine an object's stable poses. By dynamic perturbations in the simulation, this implicitly considers the robustness of the stable pose. Given the geometry of the object, we initialize it in the physics simulation above the plane in uniformly sampled rotation intervals. The simulation is progressed until the simulated object moves less than a threshold, i.e., until a stable pose has been reached.

As this process yields many equivalent poses, we cluster them to find a small set of stable poses per object. First, we discard in-plane rotation. Next, we compute the scalar product between any two rotation quaternions, yielding an angular distance matrix. All rotations with an angular distance below a threshold are considered equivalent and will thus be represented by a single representative. This is achieved by applying the Cuthill-McKee algorithm [106] to the resulting adjacency matrix, producing a block diagonal matrix where each block corresponds to a cluster of equivalent rotations. The mean rotation and z-translation per cluster initialize one stable pose hypothesis, as illustrated in Figure 4.2.

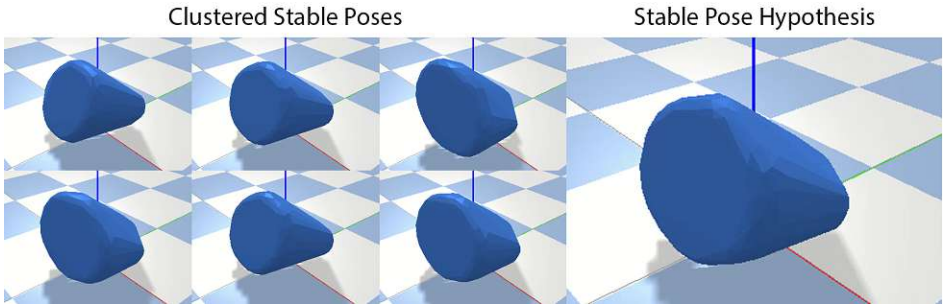


Figure 4.2: Stable pose clustering using the convex hull of the *duck* model. Equivalent poses (left) are clustered and averaged to yield a stable pose hypothesis (right).

4.3 Pose Estimation by Verification

To compute an object pose estimate using the stable poses, we have to estimate the in-plane rotation and in-plane translation from the observation. The pose estimate should minimize the discrepancy between the observation of the object and a rendering under the estimated pose. We achieve this by (1) taking an initial guess at the in-plane translation using the observed depth, (2) generating a set of in-plane rotation hypotheses for each stable pose, (3) adapting the translation estimate per rotation hypothesis and (4) selecting the estimate with lowest discrepancy based on a rendering-based verification score. These steps are outlined in Figure 4.3 for two stable pose hypotheses and one in-plane rotation hypothesis each.

The centroid of the segmented point cloud gives a translation estimate of the object's surface. But the offset of the surface from the object's reference frame, in general, depends on the rotation of the object. To estimate the in-plane rotation, we generate a set of rotation hypotheses for each stable pose of the detected object. This allows the initial translation estimate to be adapted with respect to the object rotation. We render the object under each hypothesis and compute the surface translation. The in-plane translation estimate for each

hypothesis is then corrected by the in-plane offset of the observed and the rendered surface. This improves the alignment as shown in Figure 4.3 (mid). For each hypothesis in the resulting hypotheses pool, we compute a rendering-based verification score as in Equation (3.1). The verification score considers the discrepancy between the observed and estimated object surface in terms of depth and surface normals. The selected pose estimate is the one that best fits the observed scene, for example, the bottom one in Figure 4.3.

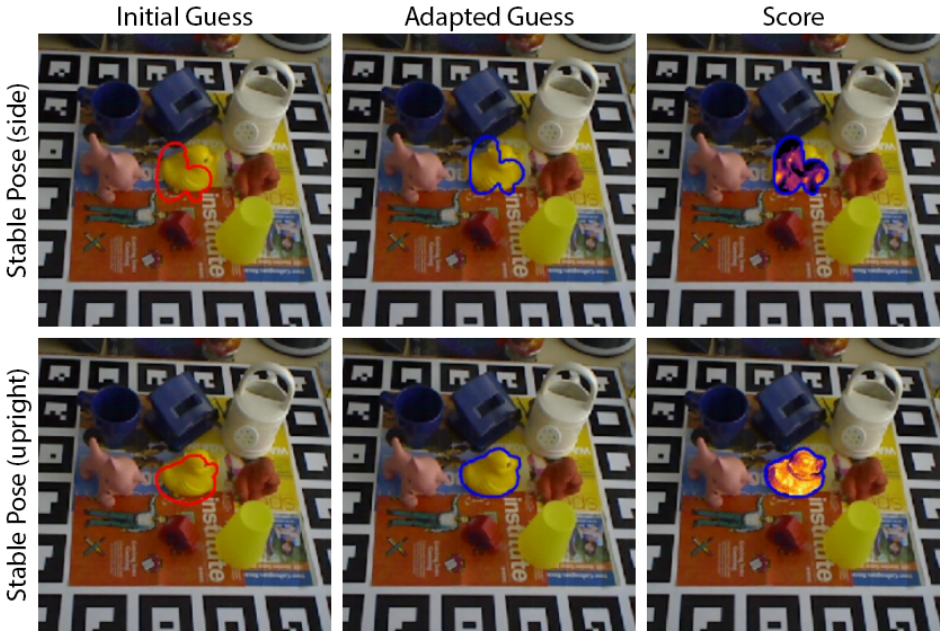


Figure 4.3: Stable pose verification. The initial in-plane estimate (left) is adapted (mid) and verified (right). The stable pose hypothesis with the *duck* model on its *side* (top) is less plausible than the *upright* hypothesis (bottom).

The proposed approach solely depends on non-textured object meshes for pose estimation. Given a detector that reliably detects novel instances of known classes, it is therefore straightforward to adapt the resulting scene explanation pipeline to changing environments.

4.4 Experiments

We evaluate how the proposed approach compares to state-of-the-art pose estimation methods that use synthetic training data and 3D models. In addition, we present the results of an ablation study on the impact of the mesh quality on the pose estimation performance.

4.4.1 Comparison to State of the Art

We compare our approach against three methods using either synthetic data or 3D models for training on the LINEMOD (LM) and OCCLUDED (OCC) datasets. Similar to our approach, the RGB-based methods by Kehl et al. [107] and Sundermeyer et al. [45] use the 2D detection to estimate the object translation. Both methods use synthetic RGB images for training, generated from precise 3D models. While [45] uses an augmented autoencoder to estimate the

object rotation in camera space, we use physics simulation and rendering-based verification to estimate the rotation in world space. The depth-based method by Vidal et al. [11] uses Point-Pair Features (PPF) and 3D models for training. For our approach, we use the highest scoring hypothesis as initial estimate and for further refinement. Note that, alternatively, multiple hypotheses as shown in Figure 4.4 could be refined and scored afterwards to potentially find a better initialization. See Chapter 5 for a corresponding experiment on OCCLUDED.

Figure 4.4: Top-5 poses on LINEMOD (left) and OCCLUDED (right).

As shown in Table 4.1, our approach provides better initial pose estimates than the related methods. With refinement, all methods are also using depth information. For our reported results, we use the Point-to-Point ICP implementation from PCL [108] for refinement, again outperforming by a large margin on LINEMOD and achieving accuracy only second to PPF on OCCLUDED, which additionally uses surface normal information.

	initial estimates			with refinement			
	SSD-6D [107]	AAE [45]	ours	SSD-6D [107]	AAE [45]	PPF [11]	ours
LINEMOD (AD$<0.1d$)	2.4	28.7	**86.5**	79.0	64.7	—	**92.4**
OCCLUDED (AR)	—	14.6	**51.5**	—	23.7	**58.2**	54.0

Table 4.1: Comparison on LINEMOD and OCCLUDED.

4.4.2 Ablation Study

The presented approach uses 3D models in physics simulation and for verification. An example of four model approximations is shown in Figure 4.5. They are generated using the decimate-collapse operation in Blender. In addition, the convex hull $\tilde{C}$ of the precise model is shown. Table 4.2 also reports the Hausdorff distance d_H between the precise model and the respective approximation, normalized to percent of model diameter and averaged over all objects in OCCLUDED. The AR score is computed using the precise model and thus only depends on the precision of the estimated pose.

The results in Table 4.2 show that our approach is able to perform well even with highly decimated meshes. The simulation produces similar pose estimates (results per row) and

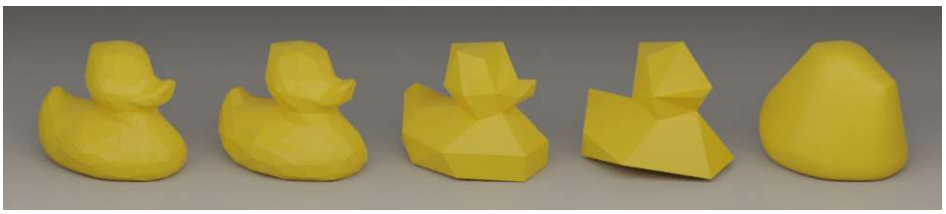


Figure 4.5: Approximate *duck* models $\tilde{O}_i$ with 704, 352, 70 and 34 faces and the convex hull $\tilde{C}$. [27].

AR		simulation $\tilde{C}_1$	$\tilde{C}_2$	$\tilde{C}_3$	$\tilde{C}_4$	$\mathbf{d_H}(\tilde{O},\tilde{O}_i)$
verification	$\tilde{O}_1$	**51.5**	50.8	48.3	49.1	0.9
	$\tilde{O}_2$	51.6	50.7	48.4	49.0	1.3
	$\tilde{O}_3$	51.4	50.4	47.8	48.4	4.3
	$\tilde{O}_4$	48.9	48.6	45.2	44.5	8.3
$\mathbf{d_H}(\tilde{C},\tilde{C}_i)$		0.9	1.1	3.4	6.6	

Table 4.2: Influence of mesh quality results in AR on OCCLUDED. The meshes are decimated with increasing index. The last row and column give the average Hausdorff distance over all objects in percent of object diameter d.

the verification is able to fit approximate meshes to the observation (results per column). The diagonal gives the results of using each approximation level for pose estimation. Note that, although the Hausdorff distance does not consider surface normal errors, it can give an estimate of the performance deterioration.

Highlights

- No training data beyond meshes is required to estimate the pose of a novel object
- Robustness to 3D model quality – a low-poly and non-textured model is sufficient
- Accuracy comparable to methods that use only synthetic data or the 3D model

Chapter 5

Object Pose Refinement and Verification using Physics Simulation and Rendering

The performance of detection and pose estimation algorithms deteriorates when the objects' 3D models are inaccurate or lighting and viewing conditions change [109], [110]. This results in misdetections and inaccurate object pose estimates. To deal with this problem, hypotheses verification and object pose refinement are commonly used in object pose estimation pipelines.

Hypothesis verification accepts or rejects (combinations of) object pose hypotheses. While this improves accuracy and reliability it also introduces the problem of increased complexity arising from the number of possible combinations in multi-object scenes. The usability of such approaches [23], [73] in robotics is thus limited by their runtime. Moreover, verification alone may not recover from generally inaccurate hypotheses. In contrast, object pose refinement improves the hypotheses themselves. This is commonly achieved by minimizing the misalignment between the observed scene and the object under estimated pose, for example, using the Iterative Closest Point (ICP) algorithm [17]. However, if the visual observation is noisy or the initial estimate is significantly inaccurate, refinement may converge to a local minimum (i.e., wrong pose) or even diverges. Alternatively, physics simulation is shown to improve accuracy and increase plausibility [23], [111] of object pose estimates. But simulation from erroneous initialization may result in objects toppling over and diverge from the true pose.

We hypothesize that, by integrating these approaches into one step, we are able to improve the overall accuracy of the pose estimates, while achieving more graceful degradation by limiting divergence of individual strategies. To this end we present VeREFINE, an integrated approach that combines hypotheses *Ver*ification, object pose *Refine*ment and physics simulation into a unified framework. The proposed method adapts to scenes of multiple objects and efficiently focuses on refining the most promising poses in multi-hypotheses scenarios.

Contributions

- Integrating refinement with physics simulation in an iterative loop
- Leveraging regret minimization to exploit promising hypotheses
- Combined into scene-level refinement and verification for multi-object scenes

The content of this chapter is based on previously published work in [24].

5.1 Integrating Hypotheses Verification with Physics-guided Iterative Refinement

In the following, we present the building blocks of our VeREFINE approach by considering increasingly complex scenarios. For individual objects, we propose an iterative physics-guided refinement loop (Sec. 5.1.1). To improve the robustness of this approach, supervision of the refinement loop through rendering-based verification is presented (Sec. 5.1.2). Given multiple estimates, a regret minimization approach is introduced to efficiently allocate refinement towards promising estimates (Sec. 5.1.3). We extend the discussed methods to consider multi-object scenes with multiple initial estimates each, where occlusion and support relationships between objects need to be considered (Sec. 5.2).

Figure 5.1: Grasping YCB-Video objects with a Toyota HSR. Initial hypotheses, shown in simulation environment (top) and computed using DenseFusion [12], are improved using VeREFINE (mid and bottom). [24].

5.1.1 Physics-guided Iterative Refinement (PIR)

Object pose refinement methods depend heavily on the quality of the initial estimates. In contrast to previous approaches that apply physics simulation as a post-hoc step after iterative refinement [23], we propose to interleave object pose refinement and physics simulation in a Physics-guided Iterative Refinement (PIR) loop, illustrated in Figure 5.2a. The physical plausibility of the initial estimate used for refinement is improved using simulation, helping the refinement to relate the correct parts of the model to the observation. The iterative feedback loop allows the refinement to, in turn, initialize physics simulation with better estimates, thus limiting divergence.

In each iteration, the current object pose estimate $\hat{T}_{cur} = [\hat{R}_{cur}, \hat{t}_{cur}]$ initializes the object in the simulation environment, shown in Figure 5.2a (bottom). In the simplest case, the environment consists of a supporting plane. In more complex scenes, it also includes other estimated objects. The simulation is progressed and the resulting object pose T_{sim} is returned. As indicated in Figure 5.2a, only the orientation part $\hat{R}_{sim}$ is used to update the estimate. This is motivated by the observation that, when physics simulation leads to large displacements, it causes the iterative refinement to lose track of corresponding object parts. We found only using the orientation contains this divergent behavior while still improving the refinement process. The estimate $[\hat{R}_{sim}, \hat{t}_{cur}]$ is used to initialize an iteration of the object pose refinement algorithm that returns the final estimate $\hat{T}_{ref}$ after one iteration of PIR. In the experiments, multiple iterations of PIR are used to obtain $\hat{T}_{ref}$.

5.1.2 Supervised Iterative Refinement (SIR)

Due to divergent behavior in physics simulation or iterative refinement, the final estimate after applying these methods might generate a worse explanation of the observation than the initial or intermediary estimates. We solve this by continuously evaluating the observation fit of the intermediary estimates. This integration of verification into the refinement process allows us to supervise divergent behavior and select the best fitting estimate as the final one. The verification score $\bar{a}$ in Equation (3.1) measures the observation fit and is computed from the average discrepancy between the estimate and the observation in terms of depth and surface normals. Figure 5.2b (bottom) shows an example of $\bar{a}$ applied to an estimate.

In each PIR iteration i, we evaluate the estimates returned by physics simulation $\hat{T}_{i;sim}$ and refinement $\hat{T}_{i;ref}$ and proceed with the estimate that achieves the better score. After the last iteration, the final estimate $\hat{T}$ that gives the best score $\bar{a}$ overall is selected from all processed estimates. As such, in cases where the individual approaches could diverge, Supervised Iterative Refinement (SIR) can recover to the highest scoring intermediary estimate.

5.1.3 Regret-minimizing Iterative Refinement (RIR)

Considering multiple pose hypotheses per object raises the questions: On which hypotheses to spend refinement steps and which hypothesis to select in the end. Promising hypotheses should be exploited by applying more refinement steps while other hypotheses should still be explored to find better candidates.

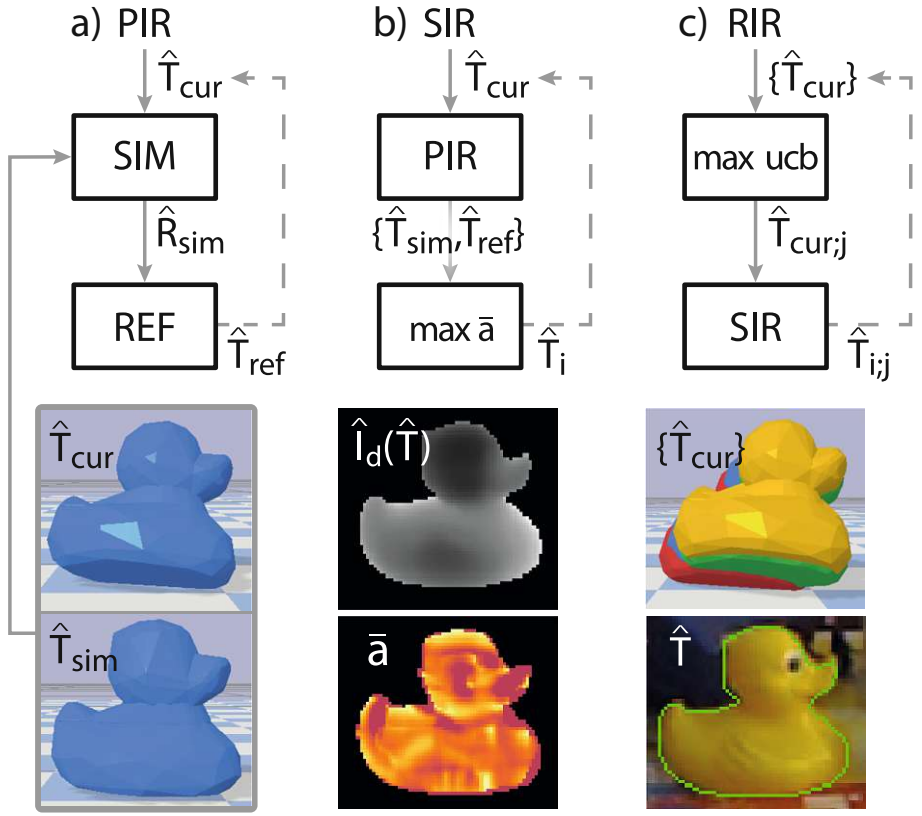


Figure 5.2: Proposed integration approaches given an initial object pose estimate (T_{cur}). (a) Integration of physics simulation (SIM) and iterative refinement (REF) into Physics-guided Iterative Refinement (PIR). (b) Supervision using verification score $\bar{a}$ (SIR). (c) Regret minimization using UCB score (RIR). [24].

We propose to use a Multi-armed Bandit (MAB) to model this exploitation-exploration problem, where the pull of arm j represents running one SIR iteration for hypothesis j. The Upper Confidence Bound policy (UCB) [112] minimizes the regret of choosing a sub-optimal arm of a MAB with respect to a given reward. In each iteration, the arm with maximal ucb_j is selected according to

$$ucb_j = \mu_j + c \cdot \sqrt{\frac{\ln p}{n_j}} \tag{5.1}$$

where μ_j is the mean reward of playing arm j, p is the total number of plays and n_j is the number of times the arm has been played. c is a parameter of the algorithm that controls the balance of exploitation and exploration. With $c > 0$, all hypotheses are eventually explored in the limit. The larger the μ of the best known hypotheses as compared to the others, the more it will be exploited. As a result, the RIR converges towards refining the best known hypothesis if its mean reward is significantly larger. However, if multiple hypotheses yield a similar reward, the RIR will alternate between them. In this case, reducing c forces RIR to exploit the best hypothesis even if the reward difference is small. The choice of c is thus dependent on the variation of hypotheses: If the underlying pose estimator yields similar hypotheses, it is beneficial to force exploitation using a small c. If the hypotheses expose high variance, a higher c and thus more exploration yields, in general, better results.

In our approach, illustrated in Figure 5.2c, the verification score $\bar{a}$ is chosen as reward function. Applying the UCB policy to the resulting reward statistics efficiently allocates a fixed refinement budget, spending more refinements on promising hypotheses while saving refinements on those that have a low $\bar{a}$. This results in the same total amount of refinements but in a regret-minimizing way. The resulting Regret-minimizing Iterative Refinement (RIR) procedure starts by ranking the initial estimates based on $\bar{a}$. For each subsequent RIR iteration, SIR is applied and the verification score is used as reward signal. As with SIR, the final selection is based on the observation fit across all encountered estimates.

The formulation based on a MAB and a rendering-based score allows our approach to be quickly applied to new datasets and can be used to extend existing and future refinement methods. In contrast, the related approach in [73] uses reinforcement learning and a CNN-based verification score regression, which need to be expensively re-trained.

5.2 Physics Simulation and Regret Minimization in Cluttered Multi-Object Scenes

In cluttered multi-object scenes, the proposed verification score and physics simulation need to deal with occlusions and support relationships. Thus, the order in which objects are considered is important. Moreover, with each of the N objects having n hypotheses, the number of combinations of hypotheses grows exponentially. To tame this problem, we discuss clustering strategies to reduce the number of combinations that need to be considered and present two approaches to efficiently evaluate the remaining search space.

5.2.1 Object Clustering and Dependency Graph

Mitash et al. [23] isolate objects that might interact based on the segmented point clouds. This reduces the number of objects that need to be jointly considered and thus the number of combinations. Furthermore, they argue that not all combinations of objects have to be considered. Instead, occlusion and support relationships between objects are used to compute a dependency list. A search tree is built from this list, where at layer i, object i is represented by all of its n hypotheses. This yields a tree of $(n^{N+1} - 1)/(n - 1)$ nodes. For a scene of 5 objects with 5 hypotheses each this produces a search tree of 3905 nodes (excluding the root node).

In contrast, we address more general scenarios by explicitly considering ambiguous dependencies, for example, the case where an object is occluded by another object but also supporting the same object. To resolve such ambiguities, we first decompose the independent clusters into support dependency lists. The first object in each support dependency list, the base object, is assumed to be in contact with the ground plane and supports the remaining objects in the list. The support dependency lists are then ordered front-to-back based on their respective base objects. Instead of using the resulting dependency list to grow a search tree using MCTS as in [23], we exploit our single-object approaches to reduce the solution space while allowing for iterative refinement on a scene level. The proposed representation requires only $N \cdot n$ nodes to represent the same search space as before – or only 25 instead of 3905 nodes in the example.

5.2.2 VeREFINE breadth (VF_b)

Given an ordering, as determined in Sec 5.2.1, we explore all object hypotheses by representing each object in the dependency list and its hypotheses using a RIR bandit. The scene is incrementally built by computing the best estimate for the considered object in the current environment. The object is added to the environment with the computed pose, allowing more accurate estimation of the next objects' poses. We call this approach of first exploring all hypotheses per object *VeREFINE breadth* (VF_b), shown in Figure 5.3 (blue). This results in N RIR bandits with n nodes each.

5.2.3 VeREFINE depth (VF_d)

An alternative approach, and to introduce a feedback loop that is missing in VF_b, is to iterate through the dependency list. For each iteration, the objects' RIR bandits are progressed only once. The best known hypotheses after each iteration are evaluated as a complete scene. The resulting scene fit is computed by $\bar{a}$ and is used as reward for selected hypotheses instead of the per-object reward. Thereby, hypotheses that contribute to a better overall scene fit are selected more often. This scene-first approach, called *VeREFINE depth* (VF_d), is illustrated in Figure 5.3 (green). As the procedure results in a changing reward distribution, the UCB policy is replaced with Discounted-UCB (D-UCB) [113]. The reward and plays statistics are discounted by a small factor each iteration, which reduces the impact of previous iterations and adapts to a changing reward distribution over time. This is shown to reduce the cumulative regret of the D-UCB policy as compared to UCB for abruptly and continuously changing reward distributions [114].

The RIR bandits are initialized using the rendering-based verification score as in the single-object scenario, acting as a heuristic in the first iteration through the dependency list to select better initial estimates. Therefore, instead of spending refinement steps to grow the search tree as in the MCTS-based approach [23], both our proposed approaches efficiently allocate refinement steps to more promising estimates.

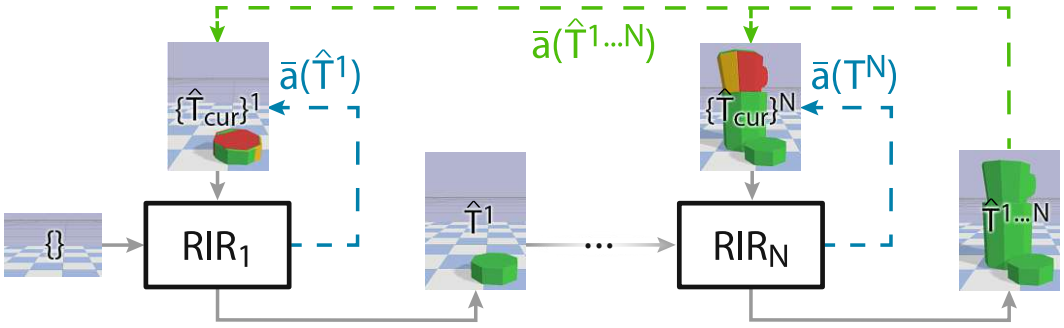


Figure 5.3: Proposed approaches for cluttered multi-object scenes. The best estimate per object is added to the simulation environment used for the subsequent objects, allowing consideration of occlusions and support relationships. VF_b (blue) fully refines each object using the object fit as reward. VF_d (green) repeats this process iteratively, refining each object only once per iteration and uses the scene fit as reward. [24].

5.3 Experiments

This section presents the evaluation of VeREFINE on the Rutgers Extended RGBD (EX), LINEMOD (LM), OCCLUDED (OCC) and YCB-Video (YCBV) datasets. Improvement over state-of-the-art refinement methods is shown by comparison with Iterative Closest Point (ICP) and DenseFusion Refinement (DF-R). For pose estimation, we use Point Pair Features (PPF) and DenseFusion (DF). In addition, we compare against the approach by Mitash et al. [23] (PHYSIM-MCTS). It uses Super4PCS (PCS) [115] and hypotheses clustering for pose estimation and Trimmed ICP (TrICP) [116] for refinement. The impact of the individual parts of our method is evaluated in an ablation study on LM.

Datasets: The LM dataset [34] is used to evaluate the single-object setting. A test set is defined based on the BOP19 challenge [31], albeit adapted to learning-based methods. These methods use the training split defined in [48], [117], [118], which excludes scenes 3 and 7 but includes 15% of the test frames used in [31]. We therefore exclude both scenes and the frames used in training from the test set for a total of 2219 test frames. EX [23], YCBV [8] and OCC [48] are used for the multi-object setting. The whole dataset is used for testing on EX. For YCBV and OCC, the test set defined in [31] is used for our evaluation.

Baselines: Mitash et al. [23] (PHYSIM-MCTS) evaluate on the EX dataset. For comparability, we use the code provided by the authors to generate bounding boxes, a pool of 25 hypotheses per object and the results reported for their method. A maximum of 150 TrICP iterations is used for evaluation of all approaches. Note that, for PHYSIM-MCTS, we only count the refinement iterations in the expansion step to ensure a fair comparison. The best performing methods on LM are the PPF-based methods by Vidal et al. [11] and Drost et al. [10]. As neither provide code, we use the code of a comparable PPF-based method by Alexandrov et al. [119] to produce a pool of hypotheses. We train Mask R-CNN [120] to provide detections and segmentation masks. In addition, we evaluate the RGB-D-based method DenseFusion [12]. It features a fast inference time and a learning-based refinement method. Precomputed detections and segmentation masks from [8] are used. A pool of object pose estimates is generated using the provided code and weights. The hypotheses pool consists of the highest confidence per-pixel estimate and additional uniformly-random sampled estimates.

We set the parameters for the verification score in Equation (3.1) to $\tau = 20$mm and $\alpha = 45$deg on all datasets. PyBullet [121] is used as physics simulator with a time-step of 1/60sec, 10 solver iterations, 4 sub-steps and assuming an equal mass of 1kg for all objects. 3D plane segmentation is employed to determine a supporting plane and its normal is used to compute the gravity direction.

The generality of our approach is shown by applying it to three baseline iterative refinement approaches, namely, TrICP, point-to-point ICP and DF-R. TrICP uses the implementation in PCL [108] with the same settings as [23]. The simulation uses 60 steps in this case. We use the basic point-to-point ICP implementation from PCL with 50 iterations. For our approaches, we distribute the ICP iterations evenly over 5 PIR iterations. DF-R uses the weights provided by [12], trained to use 2 iterations. They are distributed over 2 PIR iterations. As ICP and DF-R are found to be more sensitive to interference with the iterative refinement, only 3 simulation steps are used.

5.3.1 Ablation Study

The following ablations aim to motivate several design choices. The experiments start with the ground-truth annotations of the LM dataset as initial estimates and introduce errors of increasing magnitude. For the ablation, the ground-truth ground plane is used for physics simulation. Two types of errors are applied. (1) Rotation error is created by uniformly-random sampling a rotation axis from the unit sphere and rotating the ground-truth estimate by a varying angle about this axis. (2) Translation error is introduced by offsetting the ground truth by a translation vector that is sampled from the unit sphere, scaled by a varying distance.

5.3.1.1 Physics Simulation and Iterative Refinement

As shown in Figure 5.4 (top), our interleaved approach to combine physics simulation with refinement (PIR) is consistently the best performing simulation approach under rotation error. For translation error, it is limited as it only considers the rotation part from simulation to contain divergence. The benefit of using only rotation is illustrated by comparison with applying full simulation after refinement (PhysAfter) as used in [23]. Rotation error in the initial estimate causes this approach to diverge and perform even worse than the baseline method (DF-R) without physics simulation. Figure 5.4 (top) also shows the benefit of supervising the refinement process. Our approach (SIR) consistently improves the accuracy of pose estimates, most notably under translation error.

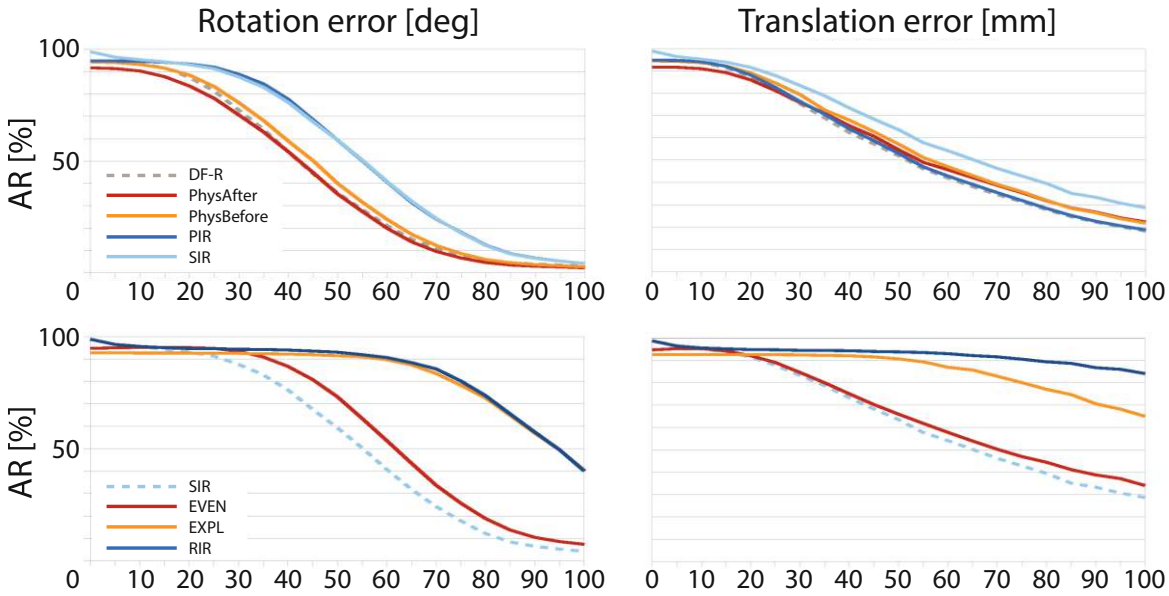


Figure 5.4: Ablations on LM using single hypotheses (top) and 5 hypotheses (bottom). EVEN and EXPL use our verification score to determine the best estimate and PIR for refinement. PhysBefore and PhysAfter apply simulation before and after refinement. AR values at 5mm/deg steps are reported and linearly interpolated in between. [24].

5.3.1.2 Regret Minimization

There are two major approaches to deal with multiple hypotheses. The first is to score all initial hypotheses, exploiting only the best scoring hypothesis for refinement (EXPL). The second approach is to refine all hypotheses evenly and selecting the best scoring refined hypothesis (EVEN). As shown in Figure 5.4 (bottom), EVEN performs well for low error magnitudes while EXPL is robust to high error magnitudes. Our regret-minimizing approach (RIR) balances between these two extreme approaches and is thus able to outperform the alternatives. Moreover, a comparison with SIR shows the benefit of considering multiple hypotheses.

5.3.2 Robustness Analysis

To highlight the robustness of our approach, we perform experiments with missing depth values to consider two types of errors. (1) Occlusion is simulated by removing rectangular patches that are centered at uniformly-random sampled positions of the observed object. (2) Missing parts of objects from the depth channel, e.g., due to reflective material, are considered by removing depth values that correspond to the object above a certain height. Error is introduced similar to the ablation study but kept fixed at 5mm for translation and 5deg for rotation. The depth error increases from 0 to 90%.

As shown in Figure 5.5, our approach increases robustness to both types of error in comparison to the baseline. This indicates that the remaining depth information, together with physics simulation, limits the degradation of performance.

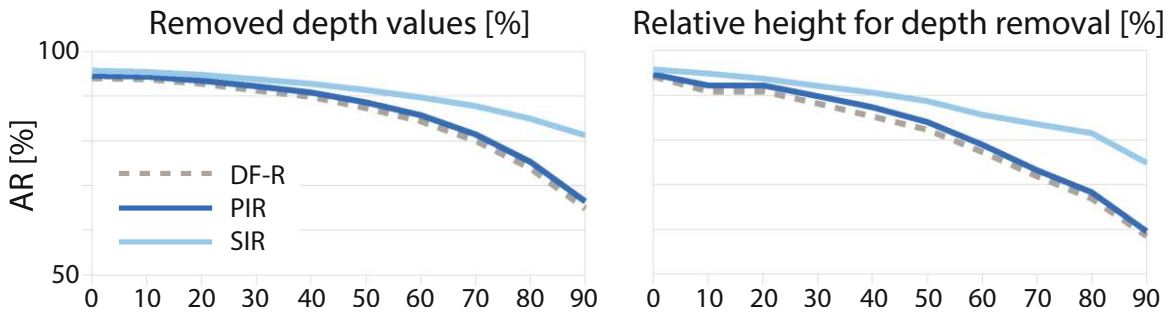


Figure 5.5: Robustness study on LM using single hypotheses with a fixed error magnitude of 5mm and 5deg. AR values are measured every 10% and linearly interpolated in between. [24].

5.3.3 Comparison to State of the Art

We compare our proposed refinement and verification approaches to state-of-the-art methods on datasets of increasing complexity, featuring single and multi-object scenarios, clutter and object interactions.

5.3.3.1 Single-Object Scenario

The single-object scenario is evaluated on the LM dataset using DF and PPF as object pose estimators and DF-R and ICP as refinement methods. The refiners are run for the same number of iterations for comparison with RIR. Results are shown in Table 5.1.

The performance of PIR indicates that physics simulation is beneficial given less accurate initial estimates using DF as compared to PPF. This agrees with our hypothesis that simulation improves implausible initial estimates while being vulnerable to divergence in inaccurate simulation environments. The biggest relative improvement is achieved by SIR, improving over DF-R by 7.1% AR. As indicated by the results using PPF, SIR is able to limit the divergence of the physics simulation observed for PIR. The top-performing approach in both conditions is RIR, improving over the baselines using the same number of refinement iterations by 5.1% and 1.3% AR, respectively. A qualitative example is shown in Figure 5.6.

DF	VSD	MSPD	MSSD	AR	T[ms]	#ref/obj
DF-R [12]	70.6	76.8	77.7	75.0	2	2
PIR	73.3	79.3	80.1	77.6	4	2
SIR	74.0	85.9	86.4	82.1	14	2
DF-R [12]	76.9	82.6	82.9	80.8	11	10
RIR	**78.3**	**89.7**	**89.6**	**85.9**	48	10

PPF	VSD	MSPD	MSSD	AR	T[ms]	#ref/obj
ICP [17]	79.8	93.2	93.0	88.7	248	50
PIR	78.1	92.1	92.1	87.4	274	50
SIR	79.9	93.7	93.2	88.9	302	50
ICP [17]	80.0	93.4	93.2	88.9	617	150
RIR	**81.0**	**95.1**	**94.5**	**90.2**	892	150

Table 5.1: Comparison using DF-R[12] and ICP[17] on LINEMOD.

Figure 5.6: Qualitative example on LINEMOD using DF-R [12] (left) and RIR (right).

Regarding runtime, we observe the application of physics simulation results in a small relative increase per frame of 1ms per simulation. Note that, using DF-R as refinement method, SIR and RIR still achieve 71fps and 21fps. The significantly improved accuracy more than outweighs the increase in runtime in the final robotic application, as shown in the grasping experiments (Sec. 5.3.4).

5.3.3.2 Multi-Object Scenario

An evaluation on the YCBV, EX and OCC datasets highlights the performance in multi-object scenarios. For comparison with RIR, VF_b and VF_d, the baseline DF-R is also run for the same number of iterations.

The results on YCBV are shown in Table 5.2. The supervision through SIR is again the biggest source of relative improvement as compared to DF-R with an increase of 2.6% AR. The increased number of refinement iterations decreases the performance of DF-R. This could be due to the confidence score of DF suggesting a sub-optimal initial estimate for exploitation or due to divergence of the refinement method itself. In either case, RIR does not exhibit divergent behavior and is able to outperform the baseline method given the same number of refinement iterations by 6.5%. Table 5.3 shows that on EX, the performance of RIR improves over the approach by Mitash et al. [23] on the VSD and MSPD metrics by 3.3% and 0.4% and significantly speeds-up the runtime.

DF	VSD	MSPD	MSSD	AR	T[ms]	#ref/obj
DF-R [12]	74.2	69.9	77.6	73.9	17	2
PIR	74.9	70.8	78.2	74.7	20	2
SIR	76.5	72.9	80.2	76.5	49	2
DF-R [12]	71.2	66.3	75.6	71.0	71	10
RIR	77.9	73.9	80.6	77.5	228	10
VF_b	78.3	73.8	80.6	77.6	495	10
VF_d	**78.5**	**74.1**	**80.9**	**77.8**	521	10

Table 5.2: Comparison using DF-R[12] on YCB-Video.

PCS	VSD	MSPD	MSSD	AR	$\bar{r}$ [deg]	$\bar{t}$ [cm]	T[s]
PHYSIM-MCTS [23]	48.5	51.6	68.3	56.2	**5.7**	1.3	29.9
RIR	51.8	52.0	63.0	55.6	10.5	1.4	5.5
VF_b	54.4	54.3	66.7	58.5	8.0	**1.2**	5.5
VF_d	**56.7**	**57.3**	**69.6**	**61.2**	7.5	**1.2**	6.2

Table 5.3: Comparison with PHYSIM-MCTS [23] using Trimmed ICP [116] on Rutgers Extended RGBD with 150 iterations each.

The results on both datasets show that our scene-level approaches successfully deal with the occlusion and support relationships in multi-object scenarios. Both VF_b and VF_d outperform [23] by a significant margin of 2.3% and 5.0% AR, respectively, with VF_d performing the best overall. A qualitative example on EX is shown in Figure 5.7. All our approaches are approximately five times faster, with TrICP accounting for 5s per frame with VF_d. This highlights the benefit of the initialization of the solutions, the efficient search space formulation and our GPU-based computation of the verification score.

As YCBV contains highly cluttered scenes that introduce occlusion but features only few support relationships, the relative increases over RIR are less pronounced with 0.1% and 0.3%. Overall, our scene-level approaches perform best on YCBV with VF_d achieving an increase of 6.8% over DF-R given the same number of iterations. A qualitative example on YCBV is shown in Figure 5.8.

Figure 5.7: Qualitative example on EX using PHYSIM-MCTS [23] (left) and VF_d (right).

Figure 5.8: Qualitative example on YCB-Video using DF-R [12] (left) and VF_d (right).

Combination with Stable Pose Estimation:

We additionally evaluate a combination of the stable pose estimation proposed in Chapter 4 and VeREFINE to determine whether the consideration of plausibility in both steps, initial estimation and refinement, benefits pose accuracy. As baseline, we use Point-to-Plane ICP [122] as implemented in Open3D [123].

As shown in Table 5.4, the baseline may not further improve accuracy beyond a certain number of iterations. In contrast, the regret-minimizing refinement of our RIR approach may explore additional initial hypotheses and thus still benefits from a larger refinement budget. Note that, for initialization, stable pose hypotheses are evaluated sequentially – scoring multiple hypotheses in parallel would significantly reduce the runtime for this step.

	Stable	P2Pl [122]	PIR	SIR	P2Pl [122]	PIR	SIR	RIR	P2Pl [122]	PIR	SIR	RIR
VSD	37.9	49.9	49.9	50.5	50.3	50.3	51.1	53.1	50.4	50.3	51.3	53.5
MSPD	60.7	66.4	66.6	67.0	66.7	67.1	67.5	71.7	66.8	67.1	67.6	72.0
MSSD	56.3	63.0	63.2	63.6	63.5	63.8	64.2	68.0	63.5	63.8	64.3	68.5
AR	51.6	59.8	59.9	60.4	60.2	60.4	60.9	64.3	60.2	60.4	61.1	64.7
T[s]	6.95	0.23	0.23	0.37	0.82	1.22	1.59	1.27	1.84	2.20	3.04	2.7
#ref/obj	–	10	10	10	50	50	50	50	100	100	100	100

Table 5.4: Comparison on OCCLUDED using Stable Pose Estimation (Chapter 4, gray) for initialization and Point-to-Plane ICP (P2Pl) [122] for refinement.

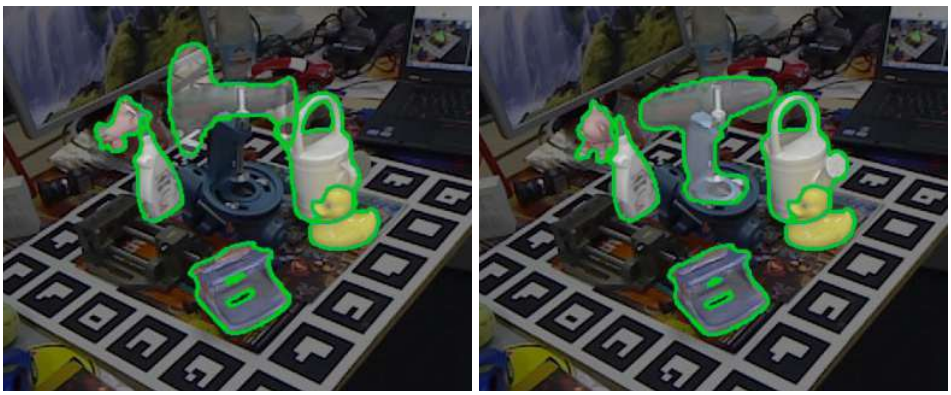


Figure 5.9: Point-to-Plane ICP [122] (left) and RIR (right) on OCCLUDED.

5.3.4 Robotic Grasping Experiment

Our work is motivated by the performance deterioration of object detection and pose estimation methods when deployed on robots [109], [110]. To evaluate whether the proposed approach is able to reduce this problem, its performance is evaluated in a grasping experiment using a Toyota HSR and YCBV objects. Reproducible experimental conditions are ensured by using the GRASPA scene layouts [92] to place 5 objects as shown in Figure 5.1. 10 grasps are attempted per object and method – 5 are attempted for a given pose and an additional 5 for a rotation to a symmetric pose. Multiple grasp poses are annotated by hand for each object as shown in Figure 5.10 (mid).

Figure 5.10: Refined estimates using RIR (left), retrieved annotated grasps (mid) and successful grasp attempt (right). [24].

In each experiment, Mask R-CNN [120] is executed to detect objects and to provide instance segmentation masks. The evaluated methods are queried to compute an object pose estimate from this information and the RGB-D image. To this end, for [23] and our approaches, we generate a hypotheses pool using DF [12]. Using the resulting pose estimate per object, the annotated grasp poses are transformed to the scene. Trajectories for all collision-free grasps are planned using MoveIt [98]. If at least one plan is found, this is counted as a *found* grasp. A grasp is considered a *success*ful grasp if the plan can be executed, i.e., the object is grasped and remains stable in the robot's gripper.

As shown in Table 5.5, our proposed approach generates object pose estimates that result in more successful and reliable grasps. The most striking improvements are achieved on the "061_*foam*_brick" and "011_*banana*" objects. Due to their proximity to other objects, object poses must be accurate to allow collision-free grasps. The *banana* is the most difficult object, resulting from inaccuracy in the instance segmentation and the object's low height. The experiment runtime is dominated by grasp planning and execution. All compared settings have overall comparable execution time, therefore, the runtime difference for pose estimation, verification and refinement is insignificant for practical applications.

DF	mustard	spam	foam	jello	banana	success	found	#ref/obj
DF-R [12]	10	3	1	7	0	42%	46%	2
SIR	9	7	2	7	0	50%	70%	2
DF-R [12]	10	6	5	9	1	62%	70%	10
PHYSIM-MCTS [23]	9	10	2	6	0	54%	78%	10
RIR	**10**	**10**	**9**	**10**	**4**	**86%**	**90%**	10

Table 5.5: Results of grasping experiments in percentage of *found* collision-free grasp poses and *success*ful grasps.

Highlights

- Improved accuracy through integration of simulation and iterative refinement
- Improved robustness to noise and initialization errors through integration of verification
- Regret minimization allows to efficiently allocate a fixed refinement budget
- May be used in conjunction with classical and learning-based refinement approaches

Chapter 6

Point Cloud Registration using Imitation and Reinforcement Learning

The methods presented in Chapters 4 and 5 consider the visual and physical aspects of plausibility in isolated steps. However, for example, improving physical plausibility through simulation competes with improving visual plausibility through iterative refinement. Instead, the influence of both plausibility aspects should be dynamically adapted depending on the scene configuration and refinement state. This motivates the design of a learning-based, plausible pose refinement approach. To this end, we consider the closely related task of point cloud registration and propose a novel approach, illustrated in Figure 6.1. Its formulation enables incorporation of plausibility aspects in Chapter 7.

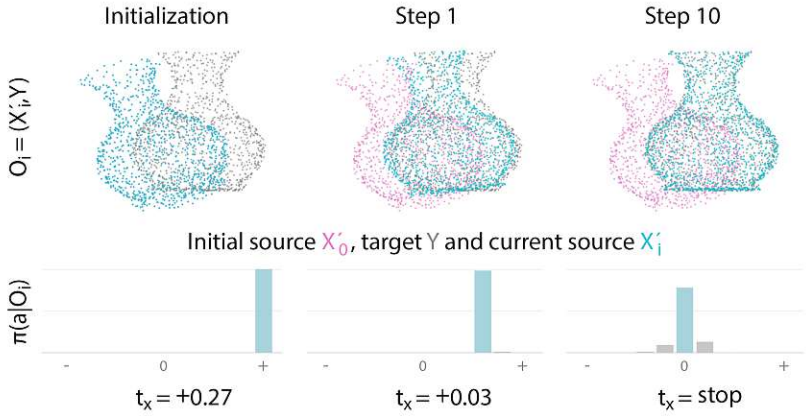


Figure 6.1: Iterative registration using ReAgent. The source point cloud (cyan) is aligned to the target point cloud (gray), starting from an initial source (magenta). ReAgent follows policy π by taking action $a_i = \arg\max_a \pi(a|O_i)$ given the current observation O_i, improving registration step-by-step. [28].

Classical registration methods such as ICP [17] generalize well to novel domains but fail when given a noisy observation or a bad initialization. Learning-based methods such as DCP [61], in contrast, are more robust but lack in generalization capacity. Depending on the application domain, these point cloud registration methods need to fulfill a range of properties. For example, robotics applications require real-time inference speed and robustness to unexpected observations. In such real-world deployment, generalization to categories that

were not seen during training is required. Further, an interaction with or scrutiny by a human might be required. For this, the method's steps need to be interpretable. These properties are often competing and thus difficult to achieve using a single approach.

In an effort to bridge this gap between methods, we design a novel registration approach that unifies accuracy, robustness to noise and initialization with inference speed. While reinforcement learning methods for RGB-based object pose refinement are proposed [15], [72], to the best of our knowledge, we are the first to consider 3D point cloud registration as a reinforcement learning problem. Our approach is based on a combination of Imitation Learning (IL) and Reinforcement Learning (RL); imitating an expert to learn an accurate initial policy, reinforcing a symmetry-invariant reward to further improve the policy. The proposed registration agent (*ReAgent*) treats registration as an iterative classification of the observed point cloud pair into discrete steps, as shown in Figure 6.1.

Contributions

- Propose a combined imitation and reinforcement learning approach to point cloud registration
- The RL-based formulation allows to adapt to different tasks and constraints by providing an adjusted expert policy or by extending the reward function
- Propose a lightweight architecture that allows fast inference

The content of this chapter is based on previously published work in [28].

6.1 The ReAgent Architecture

The proposed architecture for our point cloud registration agent is shown in Figure 6.2. The registration starts with a feature embedding that transfers the raw observed point clouds into global feature vectors. The concatenation of the source's and target's global feature vector is used as state representation, encoding the agent's information about the current registration state. A policy network uses the state representation to predict two action vectors, one for rotation and one for translation. Finally, the resulting transformation is applied to the observed source, iteratively improving the registration. In each such step, the agent receives a reward that judges how well it performs its task. The individual parts are now discussed in more detail.

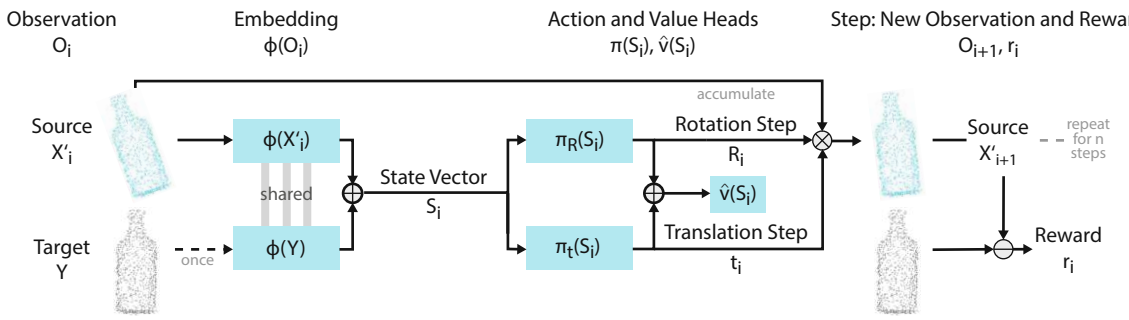


Figure 6.2: Architecture overview for one iteration of ReAgent. [28].

Learned state representation: The observed source and target may have varying shape and may be noise afflicted. Our goal is to learn a more robust and powerful representation than the bare point clouds. This is achieved by the feature embedding $\Phi(O)$, mapping from $N \times 3$ dimensional observation to $1 \times M$ dimensional state space. The source and target are passed through the embedding separately with shared weights. The concatenation of both global feature vectors is used as state S.

Discrete action space: We observe that, when trying to reach an exact registration in every iteration (i.e, by repeated one-shot registration), a bad estimate in one step may lead to divergence of the whole registration process. To this end, related work proposes to robustify the matching process [62], [63]. In an orthogonal approach, we aim to robustify the update steps themselves by using discrete, limited step sizes in each iteration. The discrete steps may be interpreted as the result of a classification of the observation into misalignment bins, as shown in Figure 6.3. Inspired by recent work by Shao et al. [15], we use a set of discrete steps along and about each axis as action space. We propose to use an exponential scale for the step sizes to quickly cover a large space during initial registration, while allowing accurate fine-registration in later steps.

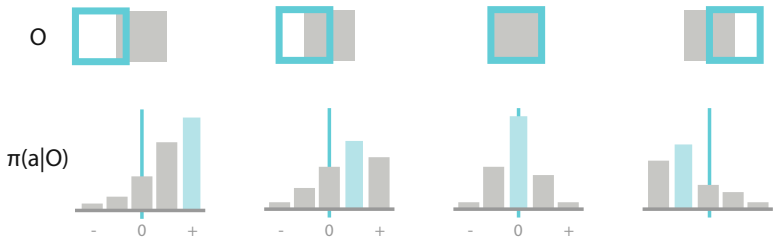


Figure 6.3: Illustration of interpretable actions. Top: Observed sources (cyan) with offset to the target (gray). Bottom: The probability of selecting each step size. [28].

Given state S, the agent's policy $\pi(S)$ gives the probability of selecting action a. The policy is computed by the agent's action head and predicts the step sizes for the iteration. Note that a is a vector of 6 sub-actions, one per rotation axis and translation axis. In addition, a value head estimates the baseline $\hat{v}(S)$. During the RL update, the baseline is subtracted from the returns of the actions to compute the advantage. This is commonly used to reduce variance as compared to using the returns directly.

Disentangled transformation: The concatenation of multiple rigid transformations with the source for iterative updates may follow different conventions. The basic approach is to compute the matrix product of all estimates $\hat{T}_i$ in homogenized form and apply this to the displaced source X'. Yet, as shown in Figure 6.4, when the rotation center is not the origin, a rotation induces an additional translation of the point cloud since

$$\begin{bmatrix} R_1 & t_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_0 & t_0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_1 R_0 & R_1 t_0 + t_1 \\ 0 & 1 \end{bmatrix}. \tag{6.1}$$

Note that this is equal to iterative application of $\hat{T}_i$ to X' as

$$R_1(R_0 X + t_0) + t_1 = R_1 R_0 X + R_1 t_0 + t_1. \tag{6.2}$$

To support interpretability, however, we would like a rotation action to only result in a local rotation of the object. Moreover, we want the rotation and translation axes to align with the

global coordinate axes such that an action in a specific axis always results in the same global displacement. Formally, we want iterative transformations to result in

$$X_i = (\prod^{i} R_i)X + \sum^{i} t_i. \tag{6.3}$$

Such disentanglement of rotation and translation not only benefits interpretability but, as shown for image-based object pose estimation by Li et al. [14], is also beneficial for training of the agent; it does not need to account for the rotation-induced translation.

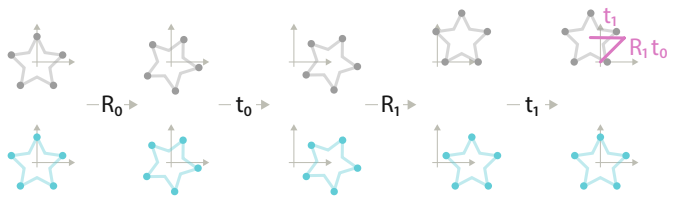


Figure 6.4: A transformation sequence and its effect on a point cloud using global (top) and disentangled transformation (bottom). [28].

Following this idea, we propose a disentangled application of $\hat{T}$ to 3D point clouds. For iterative registration, we define the update rule for an accumulator $T_i = [R_i, t_i]$ by

$$R_i = \hat{R}_i R_{i-1}, \quad t_i = \hat{t}_i + t_{i-1}, \tag{6.4}$$

which is initialized with $T_0 = [I_{3\times3},0]$ and applied to the observed source by

$$X'_i = R_i(X' - \mu_{X'}) + \mu_{X'} + t_i. \tag{6.5}$$

Thereby, rotations are applied with the centroid of the observed source $\mu_{X'}$ as the origin. Since we only apply rigid transformations, the relation between points and the centroid does not change. The outcome is that no additional translation is introduced. This also holds when applying the accumulated transformation.

6.2 Imitating an Expert Policy

Learning a complex task, such as point cloud registration, from scratch using RL may take long to converge or may get stuck with a suboptimal policy; even more so if the state representation is learned jointly with the policy. To circumvent this issue, we initialize the state representation and the policy using IL. In IL, the goal is to imitate the behavior of some domain expert. The simplest form of IL is Behavioral Cloning (BC). This assumes that, in every step, the agent has access to feedback from the expert. The feedback is used similarly to training data labels in supervised learning.

Expert policy: The expert feedback may come from interactions of a human expert or another algorithmic approach to solve the task. Since we can create training samples from point clouds by generating the initial rigid transformation, we have access to the ground-truth transformation T' between source and target. We exploit this by defining two expert policies that reduce the true transformation error in the current step, given by

$$\delta_i^R = R' R_i^\top, \quad \delta_i^t = t'_d - t_i, \tag{6.6}$$

where t'_d is the disentangled form of t' that accounts for the translation induced by R' using

$$t'_d = t' - \mu_{X'} + R'\mu_{X'}. \tag{6.7}$$

The expert policy either takes the largest possible step that reduces the *absolute* error (greedy) or the *signed* error (steady). The steady expert produces trajectories with monotonously decreasing error, while the greedy expert produces optimal trajectories at the cost of oscillation.

Data gathering: The initial transformation alone is, however, insufficient to train our agent. The agent will observe certain trajectories during inference that are not covered by the generated initial errors T'. To this end, we rollout trajectories by following the stochastic policy of the agent to gather a replay buffer. The distribution of training data in this buffer is more representative of what the agent will observe during inference and dynamically adapts as the agent improves. By using the stochastic policy, we also guarantee exploration. As the training converges, the entropy of the policy $H(\pi)$ – and hence the exploration – reduces.

Behavioral cloning: The gathered training data, together with the annotation from the expert policy, allows us to train the agent using a 6-dimensional cross-entropy loss. For every observation, we gather registration trajectories. Once a certain number of trajectories is reached, the agent is updated using mini-batches from the shuffled buffer.

6.3 Improving through Reinforcement

The resulting agent policy is in two ways limited by the expert. On one hand, the agent cannot find a better policy than the expert as its actions would differ from the expert labels. On the other hand, different transformations will, in general, lead to different expert actions. If the observed sources are, however, indistinguishable due to symmetry, they might be represented by an identical state vector. This hinders training of the agent as this would require different actions to follow from the same state vector.

Reward function: The overall goal of the agent is to align source and target. Following the expert policy, this alignment will reflect the initial transformation T'. Rather, the alignment should be treated equally for equivalent transformations $\tilde{T}'$ that result in indistinguishable observations where $X' \sim \tilde{X}'$. Instead of training the agent to exactly imitate the expert policy, we thus additionally use RL and define the training objective by a reward function.

In the proposed RL task, equal consideration of equivalent transformations is achieved by using the mean Chamfer distance (CD) between the currently observed source X' and true source $X = T'^{-1}X'$. This measure is insensitive to transformations that result in the same distance between closest points, e.g., rotations about a symmetry axis. Note, though, that the sampling rate of the point cloud may introduce fluctuations as X' moves through undersampled regions of X and vice-versa. However, this effect is lessened by considering the mean over all distances. Based on this, we define a step-wise reward

$$r = \begin{cases} -\varepsilon^-, & CD(X'_i, X) > CD(X'_{i-1}, X) \\ -\varepsilon^0, & CD(X'_i, X) = CD(X'_{i-1}, X) \\ \varepsilon^+, & CD(X'_i, X) < CD(X'_{i-1}, X). \end{cases} \tag{6.8}$$

Steps that reduce CD are rewarded by ε^+, "stop" gets a negative penalty $-\varepsilon^0$ to discourage pausing and divergent steps are penalized by $-\varepsilon^-$. We choose $\varepsilon^- > \varepsilon^+$ to discourage alternating diverging and converging steps.

Policy optimization: The policy learned using IL already performs accurate registration. Large changes to the policy due to RL might result in forgetting and thereby worsening of the agent's performance. Rather, we want the policy after an RL update to be close to the previous policy. This is achieved by trust-region approaches such as Proximal Policy Optimization (PPO) [124]. The main idea of the clipped version of PPO is to limit the ratio between the previous and the updated policy by a fixed threshold. In addition, as observed in related work combining BC and GAIL [125], it is benefitial to jointly optimize BC and RL objectives as to further limit divergence of the policy. In our combined approach, both IL and RL use the same replay buffer. Since the RL term considers equivalent transformations, the agent is able to differentiate between bad steps (discouraged by IL and RL), equivalent steps (discouraged by IL, encouraged by RL) and the best steps (encouraged by IL and RL).

6.4 Implementation Details

The final combination of IL and RL that is used to train the agent is presented in Algorithm 6.1, where *agent* implements the architecture shown in Figure 6.2.

Agent: We choose a PointNet-like architecture [90] as feature embedding. As indicated by the findings of Aoki et al. [66], the T-nets in the original PointNet architecture are unnecessary for registration and are therefore omitted. We further observe that a reduced number of embedding layers is sufficient to learn an expressive embedding. The feature embedding Φ therefore reduces to 1D convolution layers of size $[64,128,1024]$, followed by max pooling as symmetric function. The concatenation of these 1024 dimensional global features gives a 2048 dimensional state vector. In each iteration, the policy gives a step size for all 6 degrees of freedom. This is implemented as a prediction of the logits of a multi-categorical distribution. There is a total of 11 step sizes per axis: $[0.0033, 0.01, 0.03, 0.09, 0.27]$ in positive and negative direction, as well as a "stop" step. For rotation, step sizes are interpreted in radians. Shao et al. [15] propose to use shared initial layers for the action and value heads in an actor-critic design. We adapt this approach to our architecture and implement each head as fully-connected layers of size $[512, 256, D]$, where D is 33 for rotation and translation estimation and 1 for the value estimate. The concatenation of the middle layer of both action heads serves as input to the value head.

Expert: While the greedy policy achieves a lower error, when used to train the agent, both experts result in the same agent accuracy. We thus favor the more interpretable trajectories learned from the steady policy.

PPO: We use the PPO formulation from [124] for actor-critic architectures with an entropy term that encourages exploration. The advantage $\hat{A}$ in the PPO loss uses the agent's value estimate and Generalized Advantage Estimation (GAE) [126]. For the reward function, we experimentally determine $(\varepsilon^+,\varepsilon^0,\varepsilon^-) = (0.5, 0.1, 0.6)$ to successfully guide the agent.

Hyperparameters: Further parameters are the number of registration steps $n = 10$, the number of trajectories per update $N = 4$, the discount factor $\gamma = 0.99$ and the GAE factor $\lambda = 0.95$. The RL loss term is scaled by $\alpha = 2$.

Regularization: While related methods use weight decay, batch or layer normalization for regularization [61], [66], we observe that affine data augmentation achieved better results with our architecture. Namely, we use 1) random scaling sampled from $\mathcal{N}(1,0.1)$, clipped to

$[0.5,1.5]$, 2) shearing in uniformly random direction by a random angle sampled from $\mathcal{N}(0,5)$, clipped to $[-15,15]$ deg and 3) mirroring about a plane with uniformly random normal.

Algorithm 6.1 Combined Imitation and Reinforcement Learning using a Replay Buffer

for all observations O in $\mathcal{O}$ **do**
  % Gather replay buffer
  for N trajectories **do**
    for n refinement steps **do**
      agent predicts policy $\pi(O)$ and value $\hat{v}$
      action a is sampled from policy $\pi(O)$
      take action a, receive reward r and next O'
      add sample to buffer b, step observation $O = O'$
    end for
  end for
  % Process replay buffer
  compute return R, shuffle buffer b
  for all samples in buffer b **do**
    agent predicts new policy $\pi'(O)$ and value $\hat{v}'$
    % Imitate expert
    expert predicts action a^*
    compute cross-entropy loss l_{IL} of $\pi'(O)$ and a^*
    % Reinforce
    compute PPO loss l_{RL} of $\pi'(O)$ and $\pi(O)$
    % Update agent
    $l = l_{IL} + l_{RL} \cdot \alpha$
    backpropagate combined loss l
  end for
  clear buffer b
end for

6.5 Experiments

In the following, we evaluate the proposed point cloud registration agent on synthetic and real data. To evaluate our initial design goals of accuracy, inference speed and robustness, we consider noise-afflicted conditions on synthetic data. The generality of the approach is shown by results on held-out categories on synthetic data, the transfer to real data and by application to object pose refinement.

6.5.1 Point Cloud Registration on Synthetic Data (ModelNet40)

We evaluate on ModelNet40 [89], featuring synthetic point clouds sampled from CAD models.

Baselines: For comparison, we evaluate two classical and two learning-based approaches. The former are Point-to-Point Iterative Closest Point (ICP) [17] and Fast Global Registration (FGR) [59], both as implemented in Open3D [123]. PointNetLK [66] is an iterative approach based on global PointNet features. As with our approach, we set the number of iterations to 10. Deep Closest Point with Transformer (DCP-v2) [61] is a local feature-based approach, predicting one-shot registration. As the latter methods provide no pretrained models on the ModelNet40 category splits, we retrain them using the published code. All learning-based

methods (including ours) use only 3D coordinates, while the FPFH features used by FGR additionally require surface normals. On ModelNet40, the models' normals are used.

Training: All methods are evaluated using an Intel Core i7-7700K and an NVIDIA GTX 1080. We train the proposed agent using Adam [127] with AMSGrad [128] and a batch size of 32. The replay buffer contains 4 trajectories of 10 steps each, resulting in a total of 1280 observations. We pretrain the agent for 50 epochs using IL ($\alpha = 0$) on clean point clouds from ModelNet40. During pretraining, we start with a learning rate of $1e^{-3}$, and halve it every 10 epochs. We then fine-tune the policy for an additional 50 epochs on the first 20 categories of ModelNet40 with the noise defined in the following. Fine-tuning uses the same learning rate schedule, albeit starting from $1e^{-4}$. We provide separate results for training with only IL (*ours IL*) and using the combined approach (*ours IL+RL*). Note that this policy is used for all experiments and thus shows the generalization performance of the proposed approach.

Models	MAE (↓)		ISO (↓)		ADI (↑)	$\tilde{CD}$ (↓)	T (↓)
	R	t	R	t	AUC	$\times 1e^{-3}$	[ms]
ICP [17]	3.59	0.028	7.81	0.063	90.6	3.49	**9**
FGR^{+} [59]	2.52	0.016	4.37	0.034	92.1	1.59	68
DCP-v2 [61]	3.48	0.025	7.01	0.052	85.8	2.52	23
PointNetLK [66]	1.64	0.012	3.33	0.026	93.0	1.03	45
ours IL	**1.46**	**0.011**	**2.82**	**0.023**	**94.5**	**0.75**	21
ours IL+RL	1.47	**0.011**	2.87	**0.023**	94.5	**0.75**	

Table 6.1: Results on ModelNet40 with held-out point clouds from categories 1-20. Note that ↓ indicates that smaller values are better. Runtimes are for a single registration with 1024 points per cloud. $^{+}$ indicates that FGR additionally uses normals, while the remaining methods only use 3D coordinates.

.

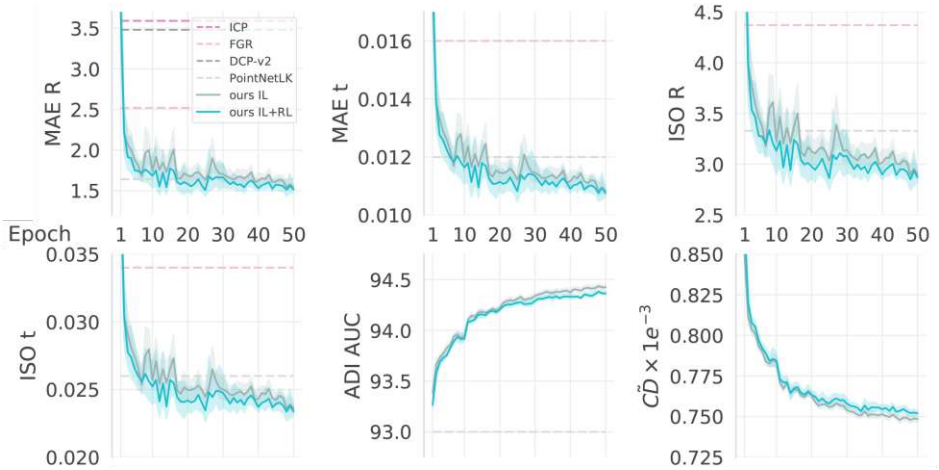


Figure 6.5: Convergence of ReAgent with 10 random seeds on held-out models of ModelNet40. The lines show the mean and the shaded areas indicate the 95%-confidence intervals. Best viewed digitally. [28].

Results: To validate generalization to unseen points clouds and novel categories, we follow related work [61], [62], [66] and use the point clouds generated by [90] based on

ModelNet40. All approaches are trained on the training split of the first 20 categories. The data augmentations follow related work [63]: Of the 2048 points, 1024 are randomly and independently subsampled for source and target to introduce imperfect correspondences. The source is transformed by a random rotation R' of $[0,45]$ deg per-axis and a random translation t' of $[-0.5,0.5]$ per-axis. Random noise is sampled (again independently for source and target) from $\mathcal{N}(0, 0.01)$, clipped to 0.05 and applied to the point clouds. Finally, the point clouds are shuffled as to permute the order of points. Table 6.1 shows results on the test split of the first 20 categories. Table 6.2 shows results on the test split of the second 20 categories. For consistency, we train the other learning-based approaches in the noisy condition.

Categories	MAE (↓)		ISO (↓)		ADI (↑)	$\tilde{CD}$ (↓)	T (↓)
	R	t	R	t	AUC	$\times 1e^{-3}$	[ms]
ICP [17]	3.41	0.024	7.00	0.051	90.5	3.08	**9**
FGR$^+$ [59]	1.68	0.011	2.94	0.024	92.7	1.24	68
DCP-v2 [61]	4.51	0.031	8.89	0.064	82.3	3.74	23
PointNetLK [66]	1.61	0.013	3.22	0.028	91.6	1.51	45
ours IL	1.38	0.010	2.59	**0.020**	**93.5**	**0.95**	21
ours IL+RL	**1.34**	**0.009**	**2.48**	**0.020**	93.3	0.99	

Table 6.2: Results on ModelNet40 with held-out categories 21-40 Note that ↓ indicates that smaller values are better. Runtimes are for a single registration with 1024 points per cloud. $^+$ indicates that FGR additionally uses normals, while the remaining methods only use 3D coordinates. [28].

.

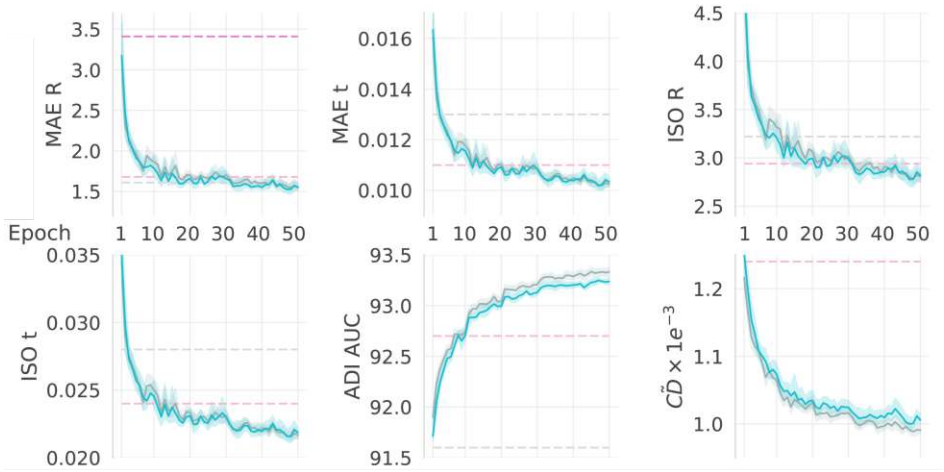


Figure 6.6: Convergence of ReAgent with 10 random seeds on held-out categories of ModelNet40. The lines show the mean and the shaded areas indicate the 95%-confidence intervals. Best viewed digitally. [28].

As shown in Table 6.2, our approach successfully generalizes to novel point clouds and novel categories. We report improved accuracy across all metrics as compared to related work. The comparison in the rightmost column shows that our approach is also the fastest of the evaluated learning-based point cloud registration methods. Inference speed is even comparable

to the one-shot method DCP-v2. However, the performance of DCP-v2 deteriorates with imperfect correspondences, as is the case with noisy observations.

When generalizing to held-out models, as shown in Figure 6.5, the addition of RL successfully improves accuracy on the rotation-based metrics. As indistinguishable observations are due to rotations in this scenario these benefit most from the policy optimization. Yet, this improvement diminishes with novel categories, shown in Figure 6.6 and also indicated by the results on ScanObjectNN. Surprisingly, the consideration of symmetry via RL even slightly decreases mean performance on ADI AUC and $\tilde{CD}$ over 10 random seeds.

Step-by-Step Results: Table 6.3 shows the results of ReAgent (IL+RL) after each iteration. T_c indicates cumulative runtime for inference using a single observation. The conditions are identical to those in Table 6.1, using held-out ModelNet40 models. The step-by-step results show that the exponential scale allows to pick large step sizes to achieve a rough alignment initially. Within about 3 steps, our approach achieves an accuracy similar to ICP in less time. Smaller step sizes are used subsequently and the performance of, for example, PointNetLK is reached after about 5 steps and 11ms of runtime. We observe that, for the last steps, ReAgent further reduces the Chamfer distance by aligning closer to indistinguishable poses – the error with respect to the true pose (indicated by MAE and ISO) increases slightly, while $\tilde{CD}$ reduces.

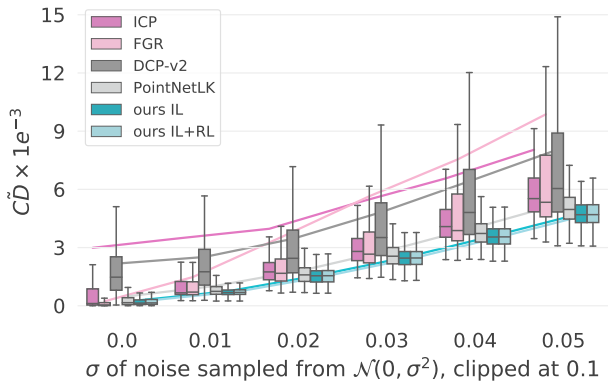
iter.	MAE (↓)		ISO (↓)		ADI (↑)	$\tilde{CD}$ (↓)	T_c
	R	t	R	t	AUC	$\times 1e^{-3}$	[ms]
init	22.35	0.238	44.49	0.478	3.4	225.6335	0
1	12.74	0.101	25.34	0.204	39.0	45.5217	3
2	6.31	0.050	12.17	0.100	69.9	10.7966	5
ICP	3.59	0.028	7.81	0.063	90.6	3.4882	9
3	3.32	0.025	6.46	0.052	83.7	2.9290	7
4	2.02	0.015	3.98	0.032	89.7	1.3366	9
PNLK	1.64	0.012	3.33	0.026	93.0	1.0305	45
5	1.50	0.011	2.97	0.024	92.8	0.9018	11
6	1.33	0.010	2.66	0.022	94.0	0.7814	13
7	1.33	0.010	2.64	0.022	94.3	0.7592	15
8	1.36	0.010	2.69	0.022	94.4	0.7528	17
9	1.40	0.010	2.76	0.023	94.7	0.7498	19
10	1.47	0.011	2.87	0.023	94.4	0.7499	21

Table 6.3: Results per iteration for ReAgent (IL+RL) on held-out ModelNet40 models. See Table 1 (left) in the main text.

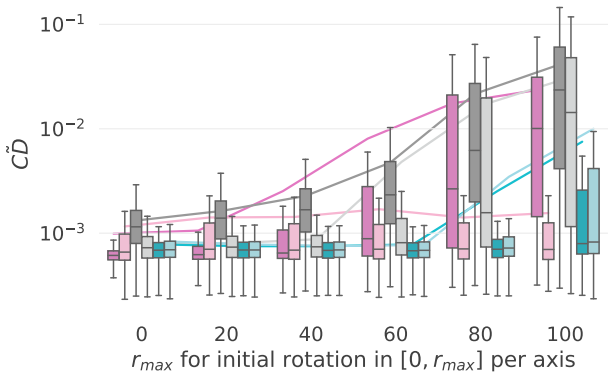
6.5.1.1 Ablation: Transformation and Noise Magnitude

To show how the compared methods are affected by varying noise and initial conditions, we provide an ablation study in Figure 6.7. We evaluate using the held-out ModelNet40 models as in Table 6.1.

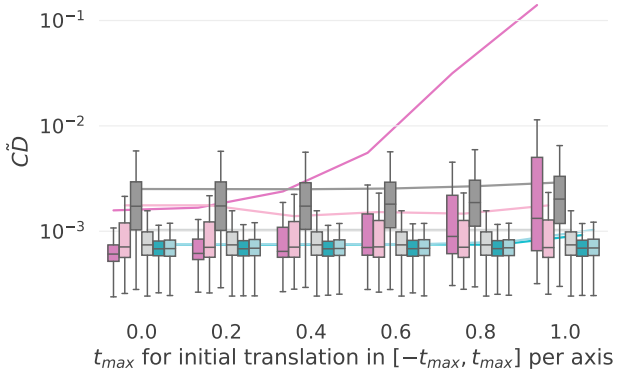
Noise Magnitude: In Figure 6.7a, the standard deviation of the distribution from which the noise is sampled is varied. The mean of the distribution remains constant at 0 and clipping remains constant at 0.1; increased from 0.05 as compared to the main experiments. FGR performs best in the noise-free condition but is heavily affected by increasing noise



(a) Varying noise magnitude.



(b) Varying initial rotation.



(c) Varying initial translation.

Figure 6.7: Results on held-out ModelNet40 models with varying noise. The boxes show $[q_{.25}, q_{.75}]$ and the median value. The whiskers indicate $[q_{.25} - 1.5IQR, q_{.75} + 1.5IQR]$, with $IQR = q_{.75} - q_{.25}$. The colored lines show the trend of the respective mean values. Note that large differences between mean and median are due to outliers. [28].

magnitude. Comparing the trend of the mean values, the other methods are similarly affected with ReAgent retaining overall best performance over all magnitudes.

Initial Transformation: In Figures 6.7b-c, the respective component of the transformation is varied, while the other is kept as in the main experiments. The magnitude of the transformation is varied by increasing the upper-bound parameter of the transformation distributions. Figure 6.7b indicates that the PointNet-based methods are able to retain high accuracy within the range of initial rotations seen during training (up to $45\,\mathrm{deg}$ per axis). FGR, using the true surface normals of the models, is barely affected by increasing rotation magnitude. In Figure 6.7c, the performance of ICP is heavily affected by the translation magnitude. This is due to a fixed upper-bound for the correspondence distance of 0.5. ReAgent also shows a slight decrease in accuracy with the highest translation magnitude, as its step sizes are limited and thus more iterations need to be spent for initial alignment.

6.5.1.2 Design of the Agent

Table 6.4 presents results for central design choices. As shown, the use of the stochastic agent policy to gather additional training data (stoch.) is essential to the success of the method. To a lesser extent, the regularization by augmenting data through affine transformations (augm.) prevents overfitting. In Section 6.2, the steady policy was already suggested to be more interpretable than the greedy policy. Even though the greedy policy on its own is more accurate than the steady policy, the results in Table 6.4 show that the agent trained using a steady policy achieves equally high accuracy. We support our choice to reduce the depth of the embedding (deep) and, instead, increasing the width of the head networks (wide) by the slightly increased accuracy.

Data		Expert		Φ, π		T		$\tilde{CD}$
stoch.	augm.	greedy	steady	deep	wide	basic	disent.	$\times 1e^{-4}$ (↓)
	✓		✓		✓		global	42.08
✓			✓		✓		global	11.81
✓	✓	✓			✓		global	2.61
✓	✓		✓	✓			global	2.92
✓	✓		✓		✓	✓		3.90
✓	✓		✓		✓		local	3.74
✓	✓		✓		✓		global	2.63

Table 6.4: Ablation study. Results of *ours IL*, pretrained on clean point clouds from held-out ModelNet40 categories.

Finally, Table 6.4 highlights the importance of the representation and application of transformations. There is only a slight improvement by using a disentangled representation with rotations applied locally ($R_i = R_{i-1}\hat{R}_i$) as compared to using homogenized transformation matrices (basic). Using globally applied disentangled rotations ($R_i = \hat{R}_i R_{i-1}$), as suggested in Section 6.1, improves both accuracy and interpretability of our agent. By using a disentangled representation, the agent does not need to account for the rotation-induced translation. With global rotations, additionally, the rotation axes remain aligned with the global coordinate axes throughout trajectories.

6.5.2 Point Cloud Registration on Real Data (ScanObjectNN)

To additionally evaluate generalization from synthetic to real data, we provide results on ScanObjectNN [91], featuring observations captured from an RGB-D sensor. We use the point clouds with segmented objects from ScanObjectNN dataset with 2048 points each. The same type of rigid transformations as in the previous condition are applied to the source. No additional noise is applied as the dataset already represents the characteristics of a specific depth sensor. For learning-based methods, the same models as in the previous conditions on ModelNet40 are used without any retraining or fine-tuning.

As shown in Table 6.5, our approach transfers from training on ModelNet40 to testing on ScanObjectNN with high accuracy. Only FGR, additionally using normals to compute FPFH features, performs consistently better under this condition. However, inference time of FGR is almost 6 times higher compared to our approach. Notably, the inference time of PointNetLK and of our approach is barely affected by the doubling of the number of points. While DCP-v2 requires repeated neighborhood computation that negatively affects inference time, both PointNet-based approaches benefit from the independent embedding per point. Qualitative examples are shown in Figure 6.8.

Segmented	MAE (↓)		ISO (↓)		ADI (↑)	$\tilde{CD}$ (↓)	T (↓)
	R	t	R	t	AUC	$\times 1e^{-3}$	[ms]
ICP [17]	5.34	0.036	10.47	0.076	88.1	2.99	**19**
FGR^{+} [59]	**0.11**	**0.001**	**0.19**	**0.001**	**99.7**	**0.16**	131
DCP-v2 [61]	7.42	0.050	14.93	0.102	72.4	4.93	54
PointNetLK [66]	0.90	0.010	1.74	0.020	92.5	1.09	45
ours IL	0.77	0.006	1.33	0.012	95.7	0.30	21
ours IL+RL	0.93	0.007	1.66	0.014	95.4	0.34	

Table 6.5: Results on ScanObjectNN with the object segmented from the observation. Learning-based methods use the model trained on ModelNet40. Note that ↓ indicates that smaller values are better. Runtimes are for a single registration and 2048 points per cloud. $^{+}$ indicates that FGR additionally uses normals.

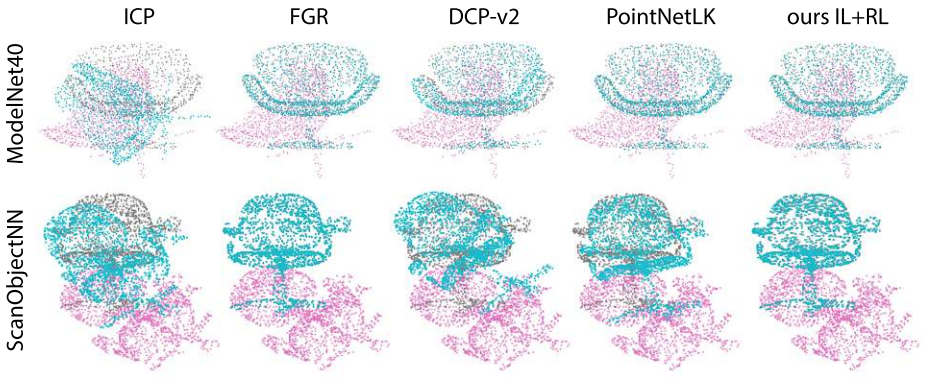


Figure 6.8: Qualitative examples. Columns show target (gray), initial (magenta) and registered source (cyan). [28].

6.5.3 Application: Object Pose Refinement (LINEMOD)

In object pose estimation, the task equivalent to point cloud registration is object pose refinement. RBy evaluation on the LINEMOD dataset [34]the performance of the presented approach in this real-world scenario is highlighted.

Baselines: We compare our method to the reported performances of RGBD (DenseFusion [61]), RGB-only (DPOD [71], DeepIM [14], PFRL [15]) and depth-only (the ICP-based multi-hypotheses method used in [8], *Multi-ICP*) object pose refinement approaches. In each block in Table 6.6, the left-most column indicates the results of the method used for initialization (gray background). With our approach, we use the estimates provided for PoseCNN [8]. As in related work [14], [15], we utilize the segmentation masks provided with the initial poses.

Training: The training phase is slightly modified on LINEMOD. Instead of using a pre-training phase, we directly use the combined approach with a learning rate of $1e^{-3}$, halving it every 20 epochs for a total of 100 epochs. The influence of the PPO loss term is reduced to $\alpha = 0.1$. Separate results for training using only IL are provided for comparison.

	RGB-based						depth-based		
class	DPOD [71]	DeepIM [14]	DPOD ref. [71]	PoseCNN [8]	PFRL [15]	DeepIM [14]	Multi-ICP [8]	ours IL	ours IL+RL
ape	53.3	78.7	87.7	27.8	60.5	77.0	79.1	**97.2**	96.9
vise	95.3	98.4	98.5	68.9	88.9	97.5	97.9	**99.6**	**99.6**
cam	90.4	97.8	96.1	47.5	64.6	93.5	93.5	99.0	**99.3**
can	94.1	97.6	**99.7**	71.4	91.3	96.5	98.7	99.6	99.5
cat	60.4	85.2	94.7	56.7	82.9	82.1	96.0	**99.8**	99.7
driller	97.7	91.6	98.8	65.4	92.0	95.0	84.2	98.8	**99.0**
duck	66.0	80.2	86.3	42.8	55.2	77.7	84.1	**96.9**	96.6
eggbox	99.7	99.7	**99.9**	98.3	99.4	97.1	98.4	99.8	**99.9**
glue	93.8	99.5	96.8	95.6	93.3	99.4	99.1	99.2	99.4
puncher	65.8	75.7	86.9	50.9	66.7	52.8	97.2	98.4	**98.6**
iron	99.8	99.7	**100.0**	65.6	75.8	98.3	90.6	97.9	97.5
lamp	88.1	98.2	96.8	70.3	96.6	97.5	94.0	**99.8**	99.7
phone	74.2	91.4	94.7	54.6	69.1	87.7	85.8	97.7	**97.8**
mean	83.0	91.8	95.1	62.8	79.7	88.6	92.1	**98.7**	**98.7**

Table 6.6: Results on LINEMOD for $AD < 0.1d$, initialized by DPOD (left) and PoseCNN (right). Symmetrical objects indicated in italics.

Results: For training, we use the split defined in related work [48], [117], [118] and sample point clouds of size 1024 per training image as source. The sampling selects a random point on the object and finds its nearest neighbors in image space. $p\%$ of the points are sampled from the object (based on the ground-truth mask) and $100 - p\%$ are sampled from the surrounding background, where p is uniformly random in $[50{,}100\%]$. Thereby, we simulate partial observation of the object and imprecise segmentation (in place of affine augmentations). As target, we uniformly random sample 1024 points from the corresponding object model. The target point cloud is normalized to be mean centered and the farthest point to be of distance 1. The same translation and scaling is applied to the source under ground-truth pose. As such, the distance from the origin provides an inductive bias on whether an (aligned) point belongs to the model or the background. Finally, we apply an uniformly

class	PoseCNN [8]	DeepIM [14]	Multi-ICP [8]	ours IL	ours IL+RL	PoseCNN [8]	DeepIM [14]	Multi-ICP [8]	ours IL	ours IL+RL
ape	5.2	48.6	38.0	70.6	**71.7**	0.0	14.3	2.9	7.5	**9.0**
vise	27.3	80.5	81.9	95.3	**96.0**	1.6	37.5	25.8	38.5	**39.9**
cam	12.5	74.0	56.1	87.7	**89.6**	0.5	**30.9**	4.2	17.8	24.8
can	26.2	84.3	81.2	95.7	**95.8**	1.0	**41.4**	10.6	39.7	41.3
cat	22.6	50.4	81.9	95.2	**95.6**	1.0	17.6	18.6	**41.6**	39.5
driller	23.7	79.2	59.3	97.1	**97.9**	1.6	35.7	5.8	46.5	**49.7**
duck	9.9	48.3	50.0	65.0	**69.4**	0.3	**10.5**	3.5	6.8	6.9
eggbox	73.9	77.8	93.1	**99.1**	98.9	17.9	34.7	**73.3**	72.4	73.2
glue	66.5	95.4	90.1	**98.7**	98.3	15.4	57.3	41.9	**76.4**	74.1
puncher	13.0	27.3	64.7	**91.3**	90.1	0.5	5.3	6.8	**31.2**	29.5
iron	23.2	86.3	60.9	**92.3**	91.5	0.7	**47.9**	5.0	34.2	34.9
lamp	29.6	86.8	85.9	98.8	**98.9**	1.6	45.3	44.0	**67.8**	66.9
phone	16.2	60.6	48.4	**90.9**	**90.9**	0.8	22.7	4.5	24.6	**25.7**
mean	26.9	69.2	68.6	90.6	**91.1**	3.3	30.9	19.0	38.8	**39.6**

Table 6.7: Per-class results on LINEMOD for $AD < 0.05d$ (left) and $AD < 0.02d$ (right), initialized using PoseCNN. Symmetrical objects indicated in italics.

random initialization error to the source, with the translation magnitude sampled from $[0,1]$ and the rotation magnitude sampled from $[0,90]$deg. During testing, we uniformly random sample 1024 points within the estimated segmentation mask and initialize the source using the estimated pose, as provided by the PoseCNN results [8].

As shown in Table 6.6, our approach outperforms all compared methods with respect to the mean AD and on most per-class results. Note that while it is more difficult for RGB methods to estimate an accurate z-translation from the camera, they more easily recover from bad segmentation masks – and vice-versa for depth-based methods. For those methods that provide results using stricter AD thresholds, we additionally provide a comparison in Table 6.7. Again, our approach increases accuracy by several percent, even achieving accuracy on the stricter $0.05d$ threshold that is competitive to the performance of the compared methods on the permissive $0.1d$ threshold. The results, moreover, indicate that the addition of RL is especially beneficial for these stricter thresholds. Since we train a single model and do not provide the agent with any information on the object class, prioritizing features that support refinement of one class might hinder that of another one. Sacrificing some generality by introducing class labels as input could thus increase performance.

For the step-wise illustrations in Figure 6.9, we zoom-in on the objects, as indicated by the black box in the top row, to highlight the high pose accuracy achieved by our ReAgent, barely distinguishable from the ground-truth pose. The pose accuracy already increases significantly within the first few ReAgent steps. For consistency, we keep using 10 steps on LINEMOD (as in the ModelNet40 experiments). While this might be reduced in real-world applications to speed-up refinement even more (below the 22ms achieved at the moment with 10 steps), the higher number of steps enables increased robustness to initialization errors.

A runtime comparison is more difficult on LINEMOD as we rely on reported numbers using different hardware. Still, the 22ms required by our approach (GTX 1080) compares favorably to 30ms for DPOD refinement (Titan X), 83ms for DeepIM (GTX 1080 Ti) and PFRL with 240ms (RTX 2080 Ti). Wang et al. [12] report no hardware and only provide

Figure 6.9: Qualitative examples on LINEMOD using ReAgent (IL+RL). As shown in the top row, 1024 points are sampled within the estimated segmentation mask. The black box indicates the zoomed-in view. Outlines for target (gray), initial (magenta) and current source pose (cyan) are shown. The last column shows a failure case. [28].

runtimes for scenes of 3 to 6 objects, with 20+10ms for DenseFusion refinement (20ms for initial embeddings) and 10.4s for the ICP-based method from [8]. The latter refines multiple hypotheses and uses rendering-based verification.

Highlights

- Improve accuracy compared to state-of-the-art methods on synthetic and real data, in registration and pose refinement
- Reduced runtime as compared to competing learning-based approaches
- Interpretable through classification into discrete refinement steps

Chapter 7

Reinforced Scene-level Plausibility for Object Pose Refinement

While the point cloud registration method presented in Chapter 6 and related depth- and RGB-based pose refinement approaches are able to increase the accuracy of initial pose estimates, they still are susceptible to ambiguity in the observation as they only consider visual alignment. RGB images provide high resolution and textural information but distance and scale are ambiguous. Using absolute depth information resolves this ambiguity. However, depth sensors offer lower resolution and suffer from artifacts such as depth shadowing and quantization errors. This is intensified by potentially erroneous instance segmentation. Moreover, indistinguishable views of an object due to (self)occlusion and symmetry lead to ambiguous refinement targets.

We propose to leverage the fact that we often observe static, rigid scenes. Thus, the objects therein need to be under physically plausible poses. To resolve ambiguous situations, approaches such as the ones in Chapters 4 and 5 additionally consider the physical configuration of the estimated scene. This additional source of information exploits the pose estimates of other scene objects, limiting the possible object interactions. Similar refinement approaches may, however, be limited to resolving collisions [75] or require physics simulation [23]. In contrast to prior work, we extend the RL-based approach in Chapter 6 and guide it towards non-intersecting, non-floating and statically stable scenes by using our contact-based definition of physical plausibility in Chapter 2. We consider surface distance as proxy for scene-level interaction information and reinforce a plausibility-based reward. Outlier removal and the consideration of geometrical symmetry further stabilize training using IL and RL.

Contributions

- Integrating physical plausibility with IL+RL-based refinement of object poses
- Simultaneously refining all objects in a scene, recovering from pose initialization errors due to ambiguous alignment using scene-level physical plausibility
- Considering geometrical symmetry and outlier points, further dealing with indistinguishable views in the depth domain

The content of this chapter is based on previously published work in [25].

7.1 The SporeAgent Pipeline

In applications such as robotic manipulation, erroneous object pose estimates may result in missed grasps. To improve upon single-shot pose estimation, object pose refinement aims to iteratively increase the alignment of an object under estimated pose with an observation. We approach this task by extending the reinforced point cloud registration approach presented in Chapter 6 towards pose refinement. This baseline approach is designed for registration of noisy yet complete point clouds. In object pose refinement, however, the observed point cloud represents only a single partial view of the object. The required instance segmentation might include points belonging to the background or close-by objects. Hence, the noisy source only partially overlaps with the noise-free target that is sampled from the object's 3D mesh. The target may also have geometrical symmetries. The correct alignment between source and target point cloud is thus ambiguous, as any symmetrical pose will result in an indistinguishable observation of the object.

To enable the RL-based registration approach from Chapter 6 to perform accurately on cluttered real data, we propose the following extensions. Consideration of geometrical symmetry in the expert policy and surface normals as additional input are discussed in Sections 7.2 and 7.3. As presented in Sections 7.4 and 7.5, we furthermore integrate contact-based physical plausibility as input and at the RL stage to deal with visual ambiguity. The resulting joint consideration of visual and physical plausibility is illustrated in Figure 7.1.

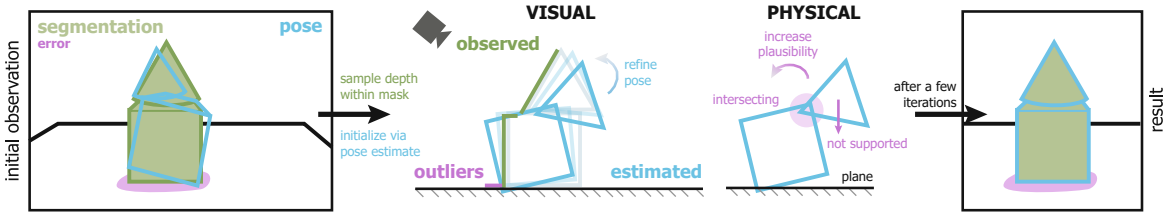


Figure 7.1: Objects are segmented and initial poses are estimated (left). The scene is initialized and all objects therein are refined iteratively. In each iteration, we more closely align the observed source and the target model point clouds (visual), while increasing physical plausibility of the scene. Adapted from [25].

7.2 Extending Input and Embedding

To deal with inaccurate segmentation of the object, we adapt the input representation and extend it by further geometrical information. Surface normals are a strong geometrical cue for outlier points as the observed normals are subject to large changes at object borders. Source normals are estimated from the observed point cloud. For the target point cloud, they are retrieved from the mesh face from which the points are sampled.

Normalizing the input point clouds provides an inductive bias that any point outside the unit sphere is either misaligned or an outlier. This also reduces the range of the potential inputs. Formally, the normalization with respect to the target Y is defined by

$$X' = (X - \mu_Y)/d_Y, \quad Y' = (Y - \mu_Y)/d_Y, \tag{7.1}$$

where μ_Y is the centroid of the target and d_Y is the maximal distance from the centroid to any point in the target. See Figure 7.2 (left) for an illustration.

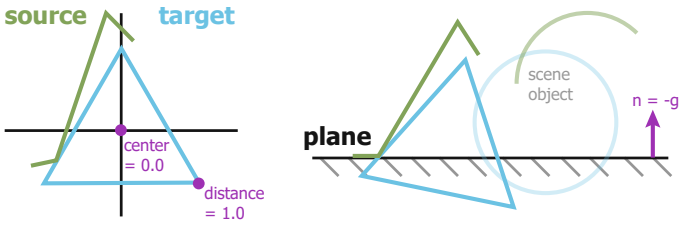


Figure 7.2: Representation for per-object refinement (left) and for computation of the scene-level information (right). [25]

Supporting the uniform coverage of the input space further, we align objects into a canonical orientation in which the major symmetry axis is aligned with the z-axis. Cylindrical, cuboid and box-shaped objects are oriented uniformly as shown in Figure 7.3. Thereby, objects' targets will more closely align and thus leverage similar geometrical features. This supports training and results in higher refinement accuracy as discussed in the ablation study in Section 7.7.3.

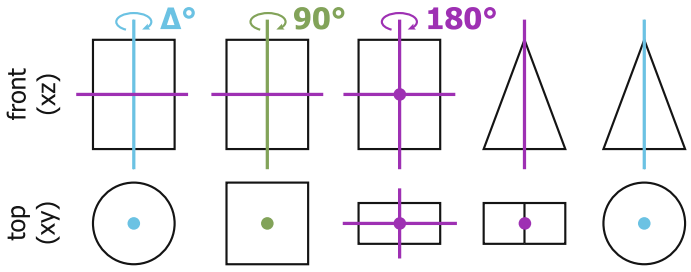


Figure 7.3: Geometrical symmetry classes. From left to right: Cylindrical, cuboid, box, front-back and rotational. Rotational symmetries (blue) are sampled with a resolution of Δdeg. [25]

To accommodate this additional input and further extensions to be discussed in the following sections, we add two additional layers to the baseline's embedding network, resulting in five 1D-convolution layers of dimension $[64,128,256,512,1024]$ in total.

Moreover, outlier points are pruned using a segmentation branch consisting of four 1D-convolution layers of dimension $[512,256,128,1]$. The concatenation of the first and last layer's feature embedding as well as a one-hot class vector results in a $1088 + k$ dimensional input, where k is the number of classes. Points that are labeled *outliers* are ignored in the max pooling operation and thus do not contribute towards the state vector. This outlier removal is trained using the ground-truth instance labels and a binary cross-entropy loss, scaled by a factor β.

7.3 Symmetry-aware Expert Policy

The expert policy presented in Section 6.2 provides the goal trajectories for IL. It is defined to minimize the residual between the estimated and ground-truth pose. However, for symmetrical objects, the object under ground-truth pose is indistinguishable from a transformation by an alternative symmetrical pose. The expert labels are thus inconsistent with respect to the observation and the agent will receive a varying loss for seemingly equivalent actions.

To address this issue, we consider multiple equivalent ground-truth poses of form $[R_s, t_s]$ for symmetrical objects. The expert should follow the shortest trajectory and thus move towards the symmetrical pose that is closest to the current estimate $\hat{T}_i$. Due to the symmetry axes coinciding with the origin in our canonical representation of the objects, the symmetry transformations do not involve any translation. The index s_i of the closest pose is thus determined by taking the residual rotation with the smallest angle

$$\begin{aligned} s_i &= \arg\min_s \arccos \frac{trace(R_s \hat{R}_i^\top) - 1}{2} \\ &= \arg\max_s trace(R_s \hat{R}_i^\top). \end{aligned} \tag{7.2}$$

The expert actions are computed as before by retrieving the symmetry-aware residuals using Equation (6.6) with respect to $[R_{s_i}, t_{s_i}]$ as ground truth.

Since most related approaches exploit RGB information, the available symmetry annotations for common pose estimation datasets [32] consider the texture of objects as well as their geometry. Hence, we need to annotate additional geometrical symmetries for use with depth-only observations. In canonical representation, this reduces to assigning objects to one of the five symmetry classes, as illustrated in Figure 7.3, in our experiments.

Visual information alone might still be ambiguous due to occlusion or observational noise. However, assuming the observed scene is static, physically plausible interactions between an object and its support limit the space of admissible object poses within the scene.

7.4 Computing Surface Distance and Critical Points

We follow the definition of physical plausibility in Section 3.2.2 as it is naturally with a point cloud-based representation. Since this approach depends on the objects' surface distances, we need to efficiently compute them for the whole scene in every refinement iteration. Moreover, we need to determine an initial supporting plane and gravity direction from the observation.

Assuming the supporting plane is sufficiently visible in the scene, we use a RANSAC-based approach to fit a plane to the subset of the source point cloud that is assigned a background label in the instance segmentation.

The surface distance is then determined as described in Section 3.2.2.4 and critical points of the queried point cloud are computed using Equations (3.7), (3.8) and (3.15), with the gravity direction given by the normal of the supporting plane. Initially, the resulting plausibility information is inaccurate, since the initial poses of the scene objects themselves are inaccurate. We therefore consider all object poses in parallel – as all poses improve, so does the scene-level plausibility information.

7.5 Leveraging Plausibility for Refinement

We want to exploit the additional source of information and guide the agent towards plausible poses. The most straightforward approach is to add the surface distance as additional input. Intuitively, the agent should move away from points that have a negative surface distance (intersecting) and keep some points close to zero distance (in contact). As the surface distance is available for both source and target, this also situates the object within the scene – if we assume two points to correspond, their surface distance under the estimated pose should also correspond.

To enforce this behavior and to also provide a training signal for the full trajectory, we add a plausibility-based reward term. To reinforce actions that lead to poses under which the object rest stably within the scene, we define

$$r_p = \begin{cases} +\rho_p, & \text{if stable,} \\ -\rho_p, & \text{otherwise.} \end{cases} \tag{7.3}$$

Note that a necessary condition for stability is feasibility. Thus, by reinforcing stability, feasibility is implicitly considered too. This reward is combined with the alignment-based reward in Equation (6.8) in a discounted task. Hence, if actions lead to a stable pose later on, the reward also reinforces these earlier actions via the discounted return.

7.6 Rendering-based Pose Scoring

Due to imprecise segmentation, the refinement may consider close-by objects or points sampled from the background. Consequently, the alignment after the final iteration might be lower than in an intermediary step. This is all the more important as poses of scene objects influence one another. Related work [13] deals with this issue by rendering-based verification of pose hypotheses.

To this end, we follow our definition of visual plausibility in Chapter 3. In each iteration, the object under currently estimated pose is rendered, giving a depth image $\hat{I}_d$ and a normal image $\hat{I}_n$. These are compared to the observed depth and normal images I_d and I_n, masked by the estimated segmentation for the corresponding object. As defined in Equation (3.1), the mean over the per-pixel scores is used to score the corresponding pose and the one with maximal score is returned as refinement result.

7.7 Experiments

We evaluate the proposed depth-based pose refinement approach in comparison with state-of-the-art methods, provide an ablation study for the proposed extensions and discuss failure cases. Qualitative examples are shown in Figure 7.4.

Datasets: The single object scenario is evaluated on the LINEMOD dataset (LM) [34].The test split defined in related work [48], [117], [118] is used, omitting scenes 3 and 7. The YCB-VIDEO dataset (YCBV) [8] features more complex scene-level interactions and heavy occlusion. In contrast to most competing approaches, we do not use any additional synthetic training data. Moreover, on YCBV, we only use 1/100th of the real training images.

For the AD metric, objects considered symmetrical are evaluated using the ADI metric and using ADD otherwise. Note that the distinction between (non)symmetrical objects in related work also considers textural information in the color image which is not observable by our depth-based approach.

Baselines: We compare our approach to state-of-the art pose refinement methods that use RGB (DeepIM [14], PoseRBPF [13], PFRL [15]), depth (Point-to-Plane ICP (P2Pl-ICP) [122], [123], Iterative Collision Check with ICP (ICC-ICP) [75], VeREFINE [24], the ICP-based multi-hypothesis approach (Multi-ICP) in [8]) or a combination of both modalities (RGBD versions of DeepIM [14] and PoseRBPF [13]). The RGB-based method of Shao et al. [15] (PFRL) is conceptionally close to our approach as it leverages RL to refine the pose by aligning the observed mask. In terms of used modality, the ICP-based approaches are considered for direct comparison. Note that the initialization by PoseCNN [8] only provides a single pose hypothesis; we indicate the use of additional pose hypotheses by *Multi-ICP* with *. P2Pl-ICP and VeREFINE (using P2Pl-ICP) are given a budget of 30 refinement iterations. The correspondence threshold for P2Pl-ICP, given as index in the results, is defined as fraction of the object size. By *(RGB)D* we indicate that a method uses depth for refinement but is initialized by a method using color information.

Implementation Details: Following related work [13]–[15], [22], we use the instance segmentation masks and poses estimated by PoseCNN [8] for initialization.

For the refinement actions, we use step sizes of $[0.0033, 0.01, 0.03, 0.09, 0.27]$ in positive and negative direction plus a "stop" action with step size 0. Step sizes are interpreted in radians for rotation and in units of the normalized representation for translation. Rotational symmetries are sampled with a resolution of 5deg. The threshold ε for computing the critical points for plausibility is experimentally determined with $1cm$. The reward function uses $\rho = (-0.6, -0.1, 0.5)$ for alignment and $\rho_p = 0.5$ for plausibility. The RL loss term is scaled by $\alpha = 0.1$ on LM and by 0.2 on YCBV. The outlier-removal loss term is scaled by $\beta = 7$. The remaining network parameters are chosen as in Chapter 6. The segmentation masks are provided as a single channel image, missing information where they overlap, and the clamps in YCBV are confused in many frames. To compute the verification score, we thus merge both masks and use thresholds $\tau_d = 2cm$ and $\tau_n = 0.7$.

During training, we generate a replay buffer of 128 object trajectories each, delaying network updates until it is filled and sampling batches from the shuffled buffer. The training samples are augmented by an artificial segmentation error, random pose error for initialization and a random error in the pose of the plane. The segmentation error selects a random pixel within the ground-truth mask. The nearest neighbors to this pixel are determined and $p\%$ of the foreground and $100 - p\%$ of the background neighbors are sampled. This simulates occlusion and inaccurate segmentation, including parts of the background. Error magnitudes are given with the experiments on the respective dataset. We train SporeAgent for 100 epochs, starting with a learning rate of 10^{-3} and halving it every 20 epochs.

7.7.1 Single Objects: LINEMOD

For training, we uniformly randomly choose the foreground fraction $p \in [50,100\%]$. The initial pose is sampled by rotation about a uniformly random rotation vector by a random magnitude in $[0, 90deg]$. Similarly, the translation is in direction of a uniformly random vector

Figure 7.4: Initial (purple) and final pose (blue) on LINEMOD (left, cropped) and YCB-VIDEO (right). [25]

of magnitude $[0.0,1.0]$ units in the normalized space of the respective object. Additionally, the estimated plane is jittered by a random rotation in $[0,5deg]$ and a random translation in $[0,2cm]$, sampled as for to the initial pose.

Table 7.1 shows the mean class recalls for varying precision thresholds with respect to the object diameter d. SporeAgent outperforms related approaches by a large margin of 3.8% using the $0.05d$ threshold and 10.8% using the $0.02d$ as compared to the next best methods.

LINEMOD	AD (↑) $< 0.10d$	AD (↑) $< 0.05d$	AD (↑) $< 0.02d$	Modality
PoseCNN [8]	62.7	26.9	3.3	RGB
PFRL [15]	79.7	–	–	RGB
DeepIM [14]	88.6	69.2	30.9	RGB
P2Pl-ICP$_{0.2}$ [123]	90.6	81.2	36.8	(RGB)D
VeREFINE$_{0.2}$ [24]	95.4	87.5	*39.5*	(RGB)D
P2Pl-ICP$_{0.3}$ [122], [123]	92.6	79.8	29.9	(RGB)D
VeREFINE$_{0.3}$ [24]	*96.1*	85.8	32.5	(RGB)D
Multi-ICP* [8]	**99.3**	*89.9*	35.6	(RGB)D
SporeAgent (ours)	**99.3**	**93.7**	**50.3**	(RGB)D

Table 7.1: Results on LINEMOD (mean over per-class results).

7.7.2 Full Scene: YCB-VIDEO

As YCBV already features large amounts of occlusion, we increase the foreground fraction p to the range $[80,100\%]$ during training. In addition to this, the training samples are more challenging and we thus reduce the initial pose error to $[0,75deg]$ and $[0,0.75]$ translation units. The same plane error as on LM is used.

The results in Table 7.2 report the Area Under the precision-recall Curve (AUC) for a varying precision threshold of $[0,10cm]$, averaged over per-class results. On the ADI metric, our approach achieves accuracy on par with RGBD-based methods, being only second to the RGBD version of DeepIM [14]. Although our approach uses depth information and, as such, may not consider textural symmetries reflected by high ADD and AD scores, we are able to achieve higher accuracy than competing RGB-based approaches. Compared to the best other evaluated depth-based approach (Multi-ICP [8]), we improve accuracy by 1 to 2.2%.

Over the best single-hypothesis depth-based approach (VeREFINE [24]), the improvement is even higher with 4.8 to 8.9%. Moreover, note that compared to the best performing learning-based approaches, we require orders of magnitude less real and no synthetic training data. A qualitative example is given in Figure 7.5.

YCB-Video	ADD (↑) AUC	AD (↑) AUC	ADI (↑) AUC	Modality
PoseCNN [8]	51.5	61.3	75.2	RGB
CosyPose [22]	–	84.5	89.5	RGB
PoseRBPF [13]	59.9	–	77.5	RGB
DeepIM [14]	71.7	81.9	88.1	RGB
PoseRBPF [13]	**80.8**	88.5	93.3	RGBD
DeepIM [14]	*80.7*	**90.4**	**94.0**	RGBD
ICC-ICP [75]	67.5	77.0	85.6	(RGB)D
P2Pl-ICP$_{1.0}$ [122], [123]	68.2	79.2	87.8	(RGB)D
VeREFINE$_{1.0}$ [24]	70.1	81.0	88.8	(RGB)D
Multi-ICP* [8]	77.4	86.6	92.6	(RGB)D
SporeAgent (ours)	79.0	*88.8*	*93.6*	(RGB)D

Table 7.2: Results on YCB-VIDEO (mean over per-class results).

Figure 7.5: Comparison of SporeAgent with competing depth-based methods on YCB-Video. PoseCNN [8] is used as initialization and shown with the depth image.

We would like to emphasize that the results reported for PoseCNN are recomputed from the poses provided by the authors of [8], as those poses serve as initialization to our method. They however differ slightly from the scores reported in their paper. Results for Multi-ICP are also computed from the poses provided by the authors of [8], [14].

Analysis of Failure Cases: We observe specific systematic failure cases that could not be resolved by considering physical plausibility. For example, as shown in Figure 7.6, the two differently sized clamps in YCBV are typically confused by PoseCNN (our initialization) and may thus end-up stuck within one another. Similarly, the bowl in YCBV may be estimated in an upside-down pose due to occlusion. In this scenario, refinement gets stuck between the plane and the scene object resting on the bowl. We hypothesize that such systematic errors can be best addressed by tighter integration of detection, pose estimation and refinement, as for example proposed by Labbé et al. [22] for CosyPose.

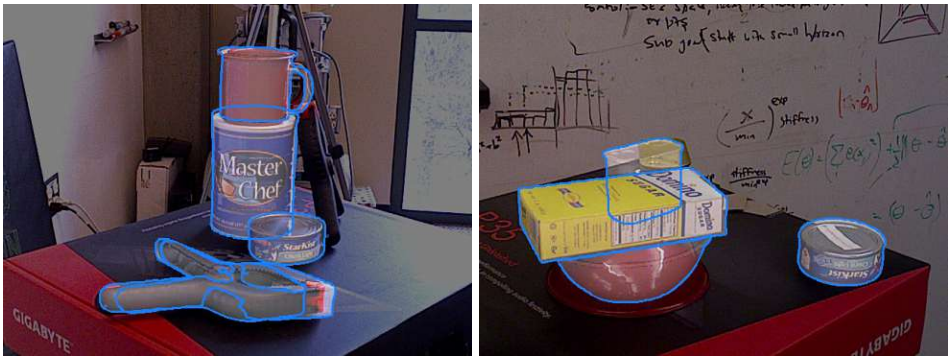


Figure 7.6: Failure cases on YCB-VIDEO. [25]

In addition, when inaccurate segmentation, occlusion or missing depth values (e.g., of metallic surfaces like caps of cans) decimate the number of foreground points in the source point cloud too much, accuracy of our approach significantly drops as the remainder of the source mostly consists of background points. Additional consideration of color information might lessen this issue.

Further robustness may be achieved by using multiple pose hypotheses per instance, as in related work [22]–[24]. For example, in the case of the confused clamp classes, one hypothesis per class might be refined and the most probable one selected using the rendering-based scoring.

Combination with Stable Pose Estimation: We additionally evaluate a combination of the stable pose estimation proposed in Chapter 4 and SporeAgent.

As shown in Table 7.3, the additional plausibility-based refinement using SporeAgent allows to increase pose accuracy by up to 2.6% on the AD AUC metric. However, illustrated in Figure 7.7 (bottom), our stable pose estimation may yield initial poses that ignore texture information that is necessary to achieve high ADD AUC and AD AUC recalls. SporeAgent may not recover from this erroneous initialization as it uses only depth and surface normal information.

	Stable	SporeAgent	PoseCNN	SporeAgent
ADD AUC (↑)	40.3	42.7	51.5	**79.0**
AD AUC (↑)	53.0	55.6	59.9	**88.8**
ADI AUC (↑)	86.3	88.5	75.2	**93.6**

Table 7.3: Comparison of using Stable Pose Estimation (Chapter 4) for initialization and using PoseCNN [8] on YCB-Video.

Figure 7.7: SporeAgent with Stable Pose Estimation on YCB-Video. Note that the stable poses (left) of the two cans in the bottom row are upside down. The refined poses (right) thus may achieve a low ADI error, while the ADD error remains high.

7.7.3 Ablation Study

To evaluate the impact of each of the proposed parts of our method, we train separate models on LM and YCBV with each part removed individually.

Table 7.4 (left) shows results on LM. We see that the scene-level information is essential to achieve high recall under restrictive thresholds. Moreover, the rendering-based scoring is able to determine the best-aligned intermediary pose – without the scoring, the last iterations jitter between tight alignment and observational noise.

LINEMOD	AD (↑) $< 0.10d$	AD (↑) $< 0.05d$	AD (↑) $< 0.02d$
SporeAgent	99.3	93.7	50.3
no normals	99.1	93.5	**50.8**
no outlier removal	99.3	93.6	50.1
no stability reward	99.3	93.7	50.0
no surface distance	**99.4**	**94.0**	46.3
no pose scoring	98.9	92.5	46.9
ReAgent (IL)	98.8	90.6	39.7

YCB-Video	ADD (↑) AUC	AD (↑) AUC	ADI (↑) AUC
SporeAgent	**79.0**	**88.8**	**93.6**
no canonical	76.4	86.7	92.6
no symmetry	78.2	88.0	93.4
no outlier removal	78.6	88.5	93.5
no stability reward	78.2	88.1	93.4
no surface distance	77.3	87.3	92.9
no pose scoring	78.3	88.1	93.0
ReAgent (IL) + normals	72.2	82.2	90.7

Table 7.4: Ablation on LINEMOD (left) and YCB-Video (right).

The ablation study on YCBV, shown in Table 7.4 (right), highlights the benefit of representing the target point clouds in a canonical frame. As on LM, the consideration of the physical

plausibility improves accuracy. The scene-level information is able to support the refinement, even when computed from initially inaccurate object poses.

Convergence with Varying Random Seed: Figure 7.8 shows the training convergence of SporeAgent on LM for 5 different random seeds. The reported AD recalls are obtained by evaluation on the test set using the current weights after the corresponding training epoch. After 50 epochs, the final performance for the $0.10d$ threshold is already achieved and the recall is within 1% for the $0.05d$ threshold. The standard deviations for the last epoch are $0.02, 0.2$ and 0.3% for the $0.10, 0.05$ and $0.02d$ thresholds, respectively.

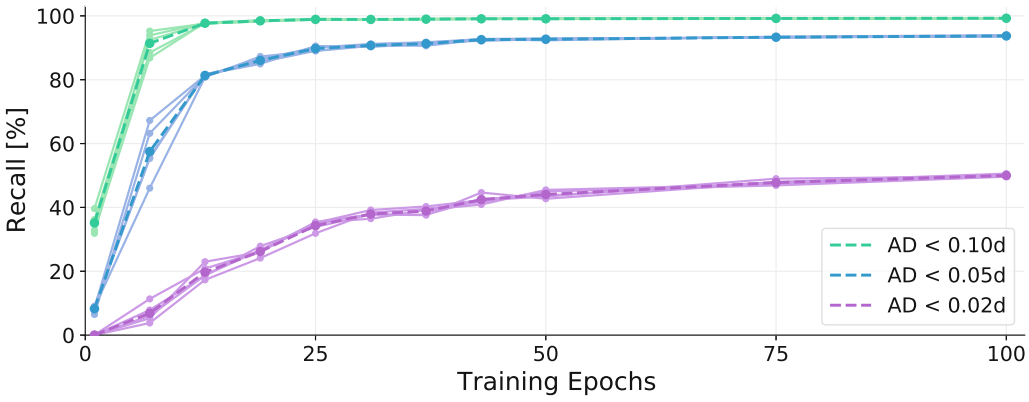


Figure 7.8: Convergence of the mean recall per epoch (dashed) on LINEMOD for 5 random seeds (solid). [25]

Number of Samples: We evaluate the impact of the number of training samples and point samples used on YCBV. As shown in Table 7.5 (left), the performance of Multi-ICP [8] on the AD AUC and ADI AUC metrics is achieved already using 1/400th of the available real training images. The number of points may be reduced to improve runtime and memory footprint while maintaining high accuracy, indicated by the results in Table 7.5 (right).

fraction	ADD (↑) AUC	AD (↑) AUC	ADI (↑) AUC
1/400th	75.4	86.2	92.5
1/200th	77.9	87.9	93.3
1/100th	**79.0**	**88.8**	**93.6**

points	ADD (↑) AUC	AD (↑) AUC	ADI (↑) AUC
256	77.9	87.8	93.3
512	78.6	88.4	93.5
1024	**79.0**	**88.8**	**93.6**

Table 7.5: Results by fraction of training split used (left) and number of points sampled for refinement (right) on YCB-VIDEO.

Number of Refinement Iterations: As indicated in Table 7.6, SporeAgent significantly improves the pose accuracy within the first few iterations. The learned policy quickly resolves large displacements by selection of the highest magnitude step size. This is also illustrated by the ablation over initial translation errors in Figure 7.9 (right). Later iterations further improve alignment.

Influence of Initial Pose: To evaluate robustness to the initialization, we apply a random error of varying magnitude on-top of the ground-truth pose. Rotation and translation errors are evaluated separately with the other kept constant at $10deg$ and 0.1 units (normalized space),

iterations	ADD AUC (↑)	AD AUC (↑)	ADI AUC (↑)
init (PoseCNN)	51.5	61.3	75.2
1	64.4	74.6	84.5
2	71.0	81.0	88.5
Multi-ICP* [8]	77.4	86.6	92.6
5	77.9	87.6	92.9
10	**79.0**	**88.8**	**93.6**

Table 7.6: Influence of number of iterations on results of SporeAgent on YCB-VIDEO. The results of PoseCNN (our initialization) and the ICP-based multi-hypothesis approach in [8] are shown (gray).

respectively. The random errors are generated as for the training augmentation. A unit vector is randomly and uniformly sampled and interpreted as rotation axis or translation direction. The corresponding rotation angle or translation distance are randomly uniform. The AD recalls for varying initial errors are shown in Figure 7.9. The results highlight the robustness of SporeAgent to translation errors. We conjecture that the normalized representation, with a centered target point cloud, simplifies the correction of solely translational offsets. Towards an error of 3 units in this normalized space, the largest step size of 0.27 (in combination with at most 10 iterations) becomes a limiting factor. Rotation errors affect performance more heavily as a partial source may be rotated to align with similar regions of the target point cloud. Yet, up to about $30deg$, the recall for a threshold of $0.10d$ is barely affected.

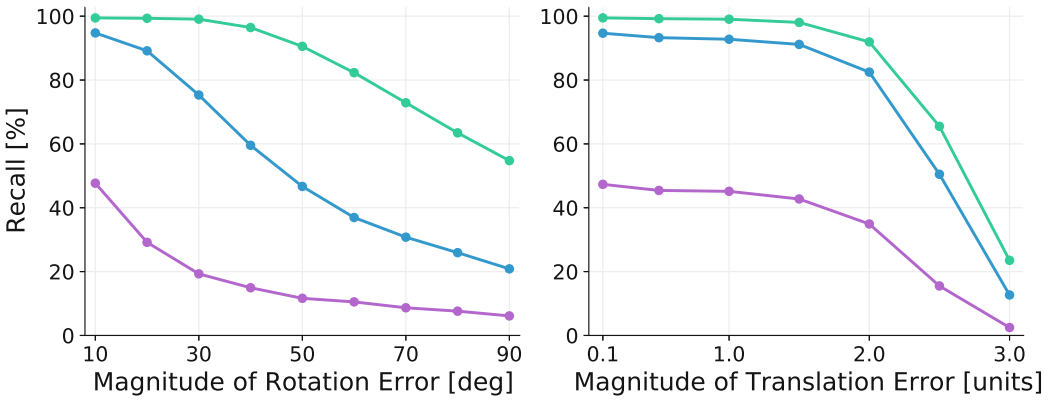


Figure 7.9: AD recalls on LINEMOD for varying pose initialization error in rotation (left) and translation (right). [25]

Visualizing the Segmentation Augmentation: The augmentation of the instance segmentation during training is visualized in Figure 7.10. The foreground (blue) and background (purple) are determined using the ground-truth visibility mask, with the background restricted to the bounding box around the mask. Both regions are pre-sampled to an equal number of points. Per sample, the augmentation randomly selects one of the foreground pixels (cross) and determines its nearest neighbors in image space. Depending on a uniformly-random fraction p and a total number of points to sample n, the $\lceil pn \rceil$ nearest neighbors in the foreground and $\lfloor (1-p)n \rfloor$ nearest neighbors in the background are sampled. This results in a coherent foreground patch, simulating occlusion or a too small mask for $p < 1$. The

background patch will consist of a part that is coherent with the foreground patch (too large mask, bleeding-out into the surrounding of the object) and a part that fades farther into the surrounding (outliers). As shown in the experiments on YCBV, where training and test scenes are different, this approach allows to learn which points to ignore rather than learning specific scene surroundings.

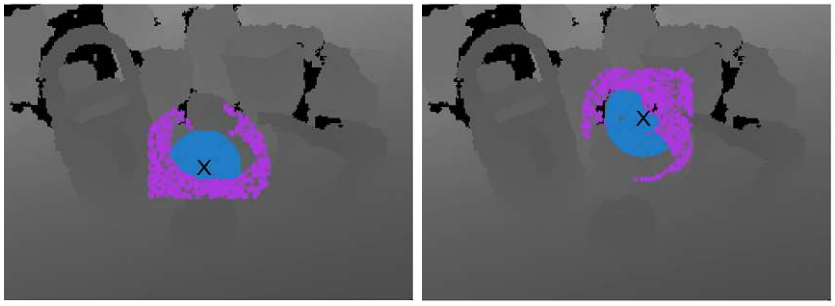


Figure 7.10: Segmentation augmentation during training with $p = 50\%$ foreground samples. Selected center (cross), sampled foreground (blue) and sampled background (purple). Background samples are limited to a bounding box around the target object. [25]

Visualizing the Scene Representation: Figure 7.11 shows the target point clouds in the scene representation for a frame in YCBV. Critical points for a queried object (the coffee can, shown in gray) are indicated. Under initial poses (left), the object would intersect with the plane and a neighboring object (shown red). The supported points (cyan) would span a supporting polygon sufficient for static stability. But we define non-intersecting and non-floating as a precondition for plausibility and hence the object pose is considered implausible under its current pose within the scene. The remaining objects in the scene are processed analogously. After refinement using SporeAgent (Figure 7.11 right), these implausibilities are resolved. Objects are resting on the supporting plane and no longer intersect (subject to slack parameter ε). Contacts (green) of the queried object with the object resting on-top of it are considered non-supported, since the surface normals of the neighboring object near the contacts are pointing in gravity direction.

Figure 7.11: Initial scene representation (left) and refined poses (right). Critical points for one target object (gray) are shown – intersecting (red), contact (green) and supported (cyan). [25]

Highlights

- Improved accuracy compared to depth-based methods, on par with RGBD-based methods
- Joint consideration of visual and physical plausibility enables robustness to inaccurate initialization in cluttered scenes
- High data efficiency as compared to competing learning-based methods, only requiring a fraction of the available real training data

Chapter 8

Explaining Plausibility Violations by Visual Alignment and Simulation

The consideration of plausibility not only serves a technical benefit but, by enabling the explanation of robotic failure, also supports users' understanding [129] of the robot's perception and actions and, thereby, fosters trust [1]–[3].

While prior work indicates interactivity and multi-modality to be useful for virtual systems [87], we integrate these explanation strategies and apply them in the context of human-robot interaction. In a proposed user study, we consider a robot failing to manipulate objects in a joint task. Interactive and single-shot explanations, visual and physical reasoning approaches are compared and evaluated in terms of their impact on the robot's trustworthiness and understandability.

Contributions

- Combining interactive and multi-modal explanations in the context of HRI
- Deriving a visual and physical reasoning strategy, based on the information available from object pose estimation
- Proposing a user study design for evaluating the impact of such explanation on users' understanding and trust

The content of this chapter is based on previously published work in [29], two pilot studies and an on-going full-scale user study. The author's contributions are the proposed visual and physical reasoning approach as well as the implementation of the virtual HRI scenario. The user study was jointly designed with Guglielmo Papagni. He proposed the interactive, multi-modal explanation approach, the questionnaires and prepared them for the online user study. Since the publication of [29], the user study design was tested in a technical pilot study, which was carried-out and evaluated in joint work with Guglielmo Papagni. Based on the technical pilot's findings, Glenda Hannibal contributed to improvements to the study by suggesting the notion of the human-robot team task and clarification of the questionnaire and explanation texts. The resulting pilot study and the on-going full-scale study are joint work of the three mentioned researchers.

8.1 User Study Design

We propose a virtual user study design based on simple user interactions and photorealistically-rendered videos of robotic manipulation. On one hand, this allows for repeatable experimental conditions between participants. On the other hand, recruitment is simplified and participation is possible without physical access to labs and robots.

During the experimental phase of the study, participants are asked to help a robot to locate and clear-away objects from a table, visualized in Figure 8.1. The Toyota HSR represents a capable embodiment for this task. Participants are told that their human-robot team may achieve up to eight points in this task, one per object. This is to give the participants "something at stake" in the interaction. They are given a description of the object that should be cleared-away next and are asked to provide the robot with an initial location. While hovering their cursor over an area around the correct object, a circle indicates the corresponding location that would be provided to the robot. Thereby, we aim to increase the perceived involvement of the participants in this human-robot interaction. After providing a location, they are shown rendered videos of the robot performing its task. The robot (and hence *the team*) initially succeeds twice. We thereby aim to reduce the effect of novelty and enable the creation of an mental model of the robot's behavior. The consistent behavior of the robot should support initial trust building [78]. Finally, the third grasp attempt of the robot fails, as shown in Figure 8.1 (bottom). This ends the experimental phase and the participants are shown different types of explanations depending on the experimental condition they are assigned to, illustrated in Figure 8.4.

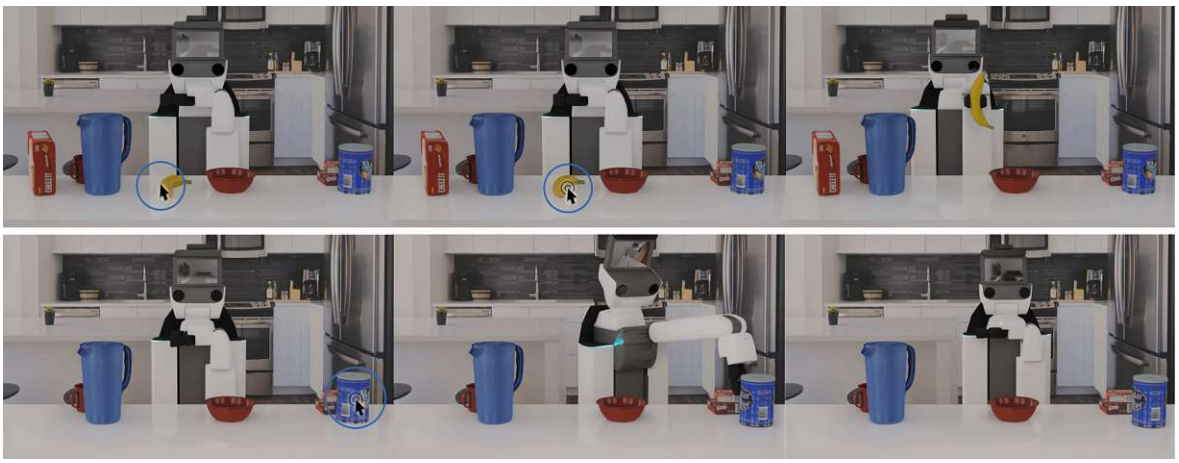


Figure 8.1: User interaction. If the user hovers the cursor above the target object, a blue circle indicates the selected area. After clicking on the object, a video of the successful (top) or failed (bottom) grasp attempt is shown.

The conditions are presented in a between-subject manner. We vary the interactivity of the explanation (*single-shot* or *interactive* with multiple layers) and the used reasoning strategy (*visual alignment of the rendering* or *displacement in the physics simulation*) of the provided explanation between groups for a 2-by-2 factorial design. Both, the single-shot and interactive groups, have access to the most detailed layer of information. The interactive group is introduced to this information in form of a dialogue with different layers of abstraction.

The study is run online and at home by each participant. After an informed consent form, we familiarize participants with the virtual robot, introducing its capabilities and showing a video of the robot moving its arm. See Figure 8.2 for an illustration. We query participants' demographic information (age, gender, first language). Participants are asked to answer items 32 and 41 of the *Need for Cognition* scale [130] as well as the 10-item *Self-Efficacy in HRI* [131] questionnaire to later test for any effects of such pre-existing biases. After this pre-experimental phase, the interaction as described above is carried out and, after the explanation, the participants are informed that their team failed. To determine the effect of the different explanatory conditions, the participants are asked to answer a questionnaire to determine the robot's trustworthiness and questions to assess their own understanding. Participants answer the 14-item *Trust Perception Scale-HRI* [132] questionnaire. They may then report whether they think that they understood the robot's explanations. If so, one multiple choice and two open text questions are used to determine whether they are able to in fact provide a sensible explanation by themselves.

Figure 8.2: Familiarization with the robot. The participants are shown a video of the robot, waving its arm to greet them. The accompanying text describes its capabilities.

We target non-expert participants as we assume them to not have an established mental model of robotic behavior and reasoning. Furthermore, non-expert end-users are the most likely to benefit from human-friendly explanations. Participants are recruited directly and through advertisement of the study. Power analysis for our 2-by-2 design and a medium effect size lead to a required sample size of 128.

8.2 Design of Visual and Textual Explanations

We aim to explain why the robot attempts to grasp the target object at a certain location by discussing its pose estimates. We show the uncertainty as well as the visual and physical plausibility of the robot's pose estimates and provide an explanation for why the robot considers the selected pose as the best.

To retain context between the experimental setting and the explanation, we overlay the proposed explanatory visualizations over the robot's scene view. The robot's uncertainty about the object's pose is expressed as multiple hypotheses that are visualized as multiple outlines,

as shown in Figure 8.3 (left). The visual plausibility of such a hypothesis is computed by our rendering-based score and visualized per-pixel in the form of a heat map. We additionally show the pose hypothesis' outline to visually link the different explanations, illustrated in Figure 8.3 (mid). Finally, in Figure 8.3 (right), we visualize the physical plausibility of a pose hypothesis by showing an additional outline for the resulting pose after simulation. The outline's color indicates the displacement between hypothesized and simulated pose.

Figure 8.3: The first layers of the interactive explanations. Uncertainty is shown by multiple outlines (left), followed the visual (mid) or a physical (right) plausibility of the best estimated pose.

We accompany each visualization with an explanatory text. To retain context between the modalities, we connect text and visualizations by re-using common identifiers, e.g., "guess #1" or the color scale indicating whether an estimate is "aligned". We use colloquial forms, such as "location" instead of pose, and abstract concepts of "estimation" and "simulation".

We design an interactive and a single-shot version of presenting these multi-modal explanations to the participants. For the interactive version, the first layer explains the robot's uncertainty, the second shows its best estimate and the final layer shows the remaining estimates for comparison. The accompanying text provide incrementally more detail in each layer and participants have the option of requesting a further explanation at layers one and two or continue with the experiment. For the single-shot version, we compile the explanatory text by retaining the information and the used keywords, while condensing the overall text as not to overwhelm participants. In addition, the visualization of the final layer is shown.

8.3 Hypotheses and Preliminary Findings

While Kulesza et al. [133] find completeness to be more beneficial for understanding than providing an overly simple explanation, there is also evidence [134], [135] that too complex explanations overwhelm users. By breaking a complete explanation down into smaller, intelligible layers, we thus conjecture our interactive conditions to provide a better understanding than the single-shot ones. The visual conditions reason with per-pixel alignment, expressed as a heat map. In contrast, the physical conditions rely on the whole object's behavior in simulation for explanation. We hypothesize that an object-based understanding is easier to interpret by non-expert users. As interpretability is fundamental to trustworthy interaction, we expect these effects to be also measurable in the perceived trustworthiness of the robot.

We carried out a pilot study with $n = 38$ participants to validate the interaction and

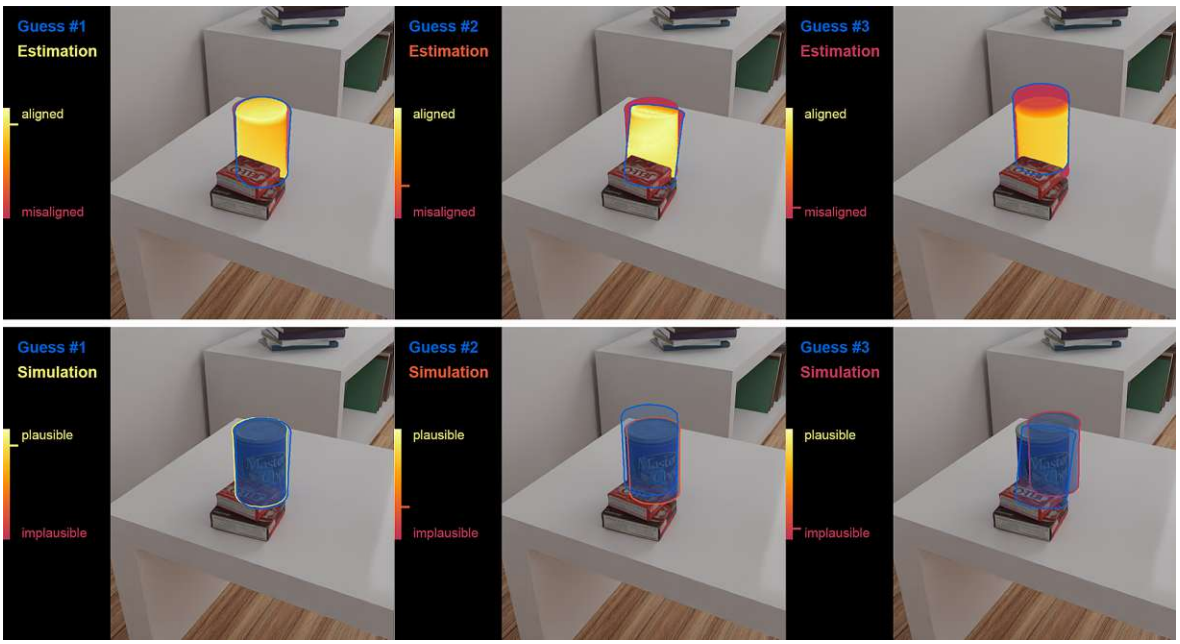


Figure 8.4: The last layers of the interactive explanations, showing the visual (top) or physical plausibility (bottom) of all estimated poses.

explanation design as well as our technical implementation. The preliminary quantitative results from this pilot study also show a tendency towards supporting the discussed hypotheses. However, we still observe high amounts of variance within the groups with such a small n (as suggested by the power analysis). Since we randomly assigned participants to the four conditions, we found an imbalance in terms of demographics, need for cognition and self-efficacy. For the on-going full-scale user study, we thus aim to manually balance the group assignment with respect to these aspects. Besides investigating our hypotheses with regards to trustworthiness and understandability, the qualitative data gathered will inform further improvement of the explanation strategies and the user study design for an eventual validation study in a real-world setting.

Chapter 9

Conclusion

A scene understanding, based on an objects' 3D models and their poses, allows a robot to explain and carry out interactions with its environment. For example, by rendering its estimated scene, it may visualize its understanding or verify its estimates. It may derive spatial and support relations between scene objects to reason how they may be manipulated or parse object-centric commands. Moreover, the robot may transform known grasp poses to the observed scene, allowing semantically-meaningful manipulation of objects such as grasping a mug by the handle.

In this thesis, we addressed challenges in object pose estimation with regards to visual alignment, physical relations and scene-level consistency of the estimated poses. We argued that these aspects influence one another – approaching them in isolation leaves ambiguous cases that result in inaccurate and conflicting estimates. The central hypothesis of this work is that both the visual and physical plausibility of the robot's scene understanding need to be jointly considered for it to handle ambiguities in its observation. While we argue that this type of reasoning is generally beneficial for perception, we focused on the specific task of object pose estimation and its sub-steps, refinement and verification.

To this end, we formulated four research questions (RQ), addressing the definition of plausibility for this task (RQ1), the joint consideration of both of its presented aspects in pose estimation (RQ2), dealing with object interactions and uncertain hypotheses (RQ3) and, finally, explanation of failure using this reasoning approach (RQ4).

9.1 Summary

In the following, we relate the chapters to the research questions, review the contributions of this thesis and summarize our findings.

Definition of Visual and Physical Plausibility of Object Poses: Addressing our first research question RQ1, in Chapter 3, we proposed a definition for visual plausibility based on the alignment of the estimated and observed scenes. By comparing rendered and camera images, this formulation accounts for (self)occlusion and extends to complex scenes of multiple objects in clutter. We show how to efficiently score object poses with respect to this definition and apply this score throughout the thesis. We present two different definitions of physical plausibility in static scenes. The first is based on dynamic physics simulation, while the second definition is based on the scene's static equilibrium. The methods we propose in

this thesis exploit these considerations to determine stable pose estimates and to iteratively refine implausible ones.

Enforcing Plausibility through Rendering and Simulation: We propose a simple object pose estimator in Chapter 4 that attempts to address RQ2 and aims to validate our plausibility definitions. Our proposed approach requires only a simple non-textured model per object to determine its physically stable poses and leverages rendering-based verification for pose estimation. This enables our approach to be quickly adapted to changing environments. Our approach presented in Chapter 5 addresses both RQ2 and RQ3 by tight integration of hypotheses verification, refinement and physics simulation. The rendering-based hypotheses verification and the proposed simulation-guided extension to iterative refinement benefit from this integration by allowing them to share plausibility information at object- and scene-level. An efficient MAB representation allows a significant reduction of runtime compared to MCTS-based approaches. The comparison with state-of-the-art methods and a robotic grasping experiment show that our integrated approach creates more accurate and more reliable object pose estimates.

Enforcing Plausibility through Learning-based Approaches: We investigate an even tighter integration of visual and physical plausibility aspects in Chapters 6 and 7 to propose a learning-based solution to RQ2. As first step, in Chapter 6 we present a novel point cloud registration agent, called ReAgent, that leverages IL and RL to achieve accurate and robust registration of 3D point clouds. Its discrete actions and steady registration trajectories provide interpretable pose refinement (RQ4), while achieving fast inference times. In Chapter 7, we extend our IL+RL-based refinement approach through consideration of the scene's static equilibrium. The resulting point cloud alignment, combined with rendering-based verification of the intermediary poses, moreover ensures the visual plausibility of the pose estimates (RQ2). Our proposed method jointly refines the poses of multiple objects in an observed depth frame in parallel (RQ3).

Explaining Plausibility Violations: Finally, in Chapter 8, we exploit the information computed by our proposed pose estimation methods to provide explanations in failure cases, providing an approach to address RQ4. In addition, we present a user study to evaluate whether these explanations allow to increase trustworthiness and understandability of a robotic interaction partner.

Review of the Research Questions and Contributions:

RQ1 – How to define the plausibility of object poses in static scenes? We defined visual plausibility as a rendering-based alignment score and, additionally, leverage point cloud-based visual plausibility in the alignment reward of the proposed registration and refinement agents. For physical plausibility, we proposed to determine stable poses using simulation, to leverage simulation for refinement and derived a contact-based definition, evaluating the scene's static equilibrium. We discussed the technical benefit of employing these definitions in object pose estimation and robotic grasping as well as their applicability in generating understandable explanations for human-robot interaction.

RQ2 – How to overcome the ambiguity of using either visual or physical reasoning in object pose estimation, refinement and verification? We showed that, by jointly considering these two aspects of plausibility, we are able to achieve increased pose accuracy in situations where either aspect alone would be ambiguous. We explored how our notion of plausibility may be combined with different reinforcement learning approaches, namely MAB, IL and

RL. We proposed a set of object pose estimation, refinement and verification approaches that are solely based on the 3D model of the objects and may be directly used to augment existing pipelines. Our proposed rendering-based verification increases robustness to partial observations, inaccurate pose estimates and divergent refinement steps. Exploiting the combined visual and physical plausibility information, we present a learning-based pose refinement method that considers intersecting and supported points between interacting objects.

RQ3 – How to efficiently consider multiple objects with multiple hypotheses? We proposed efficient representations of multi-object, multi-hypotheses search spaces through dependency lists and a MAB formulation. We used the UCB policy to balance exploration and exploitation of hypotheses. By additionally increasing the computational efficiency of steps, such as refinement and scoring, we significantly decreased the runtime in cluttered scenes. Moreover, our proposed approaches consider visual and physical scene-level object interactions.

RQ4 – How to leverage the plausibility information for explanation of robotic failure? We give an outlook to ongoing work investigating the exploitation of the plausibility information computed by our approaches to generate human-understandable explanations of robotic failure. Preliminary results indicate that this may help to increase human interaction partners' understanding of the robot and, thereby, allow robots to better retain human's trust in case of perception errors.

9.2 Outlook

To conclude, we discuss future improvements to the definitions and methods presented in this work. In addition, we propose challenges beyond the scope of this thesis and, building on our findings, potential avenues for addressing them.

The visual plausibility considerations could be extended to include color information as to deal with objects' texture and would thereby also increase the methods' robustness to partial depth data, e.g., on metallic surfaces such as caps of cans. For consideration of physical plausibility, we assumed the observed scene to be ultimately supported by a planar surface that is sufficiently visible to estimate its pose. Assuming this planar support to be horizontal, we used its surface normal as estimate of the gravity direction. However, a robot's IMU could be used instead (where available), which would also allowing to deal situations where this underlying support is non-planar.

This would require to incorporate such a-priori unknown objects in the scene representation. To this end, the use of shape estimation could be explored. Inaccuracy in the estimated shape is tolerable for this task, as long as the support surfaces are sufficiently reproduced. Alternatively, the use of mesh retrieval methods could be explored. This approach could moreover provide the input for our mesh-based pose estimation method in Chapter 4, enabling the robot to estimate the pose of novel object instances as they are encountered.

Further extension of our registration and refinement agents may require to replace the simplistic PointNet-like embedding. Embeddings operating on multiple scales, additionally using hand-crafted features or using different concepts altogether, such as graph-based approaches, would allow to increase the model capacity and hence support transfer to more complex domains. An additional improvement to the proposed agents would be to dynamically adapt the step sizes used for refinement. As observed in our experiments, the discrete steps currently

used induce finite accuracy and bound the initial error that may be overcome by the agent in a given number of iterations. Moreover, errors affecting multiple steps of the perception pipeline – such as misdetections or inaccurate segmentation – may be best addressed by tighter integration of detection, pose estimation and refinement.

A robotic prototype that uses the presented methods to generate a scene explanation and the corresponding explanations of its actions (and failures) would enable further evaluation of our approach, presented in Chapter 8, in the ever-changing environments that are inhabited by the robots' human interaction partners.

Going Beyond: Nevertheless, many objects that robots have to deal with are not yet covered by the presented methods' visual opacity, rigidity and static-scene assumptions. Dealing with articulated (or even deformable) objects, potentially being manipulated by a human hand or robotic gripper and exposing high intra-class variance in texture and shape, is beyond the scope of this work.

Reflective and refractive surfaces are typically missing or afflicted by heavy noise in depth images. As such, they may neither considered by our depth-based visual plausibility definition nor our point cloud-based refinement approaches. For simple cases, the inclusion of color information may be explored. However, specular reflection, transparency or refraction of changing backgrounds will require dedicated methods as color information becomes ambiguous in these situations. Assuming knowledge of the scene objects and their BSDF, pose refinement that leverages differentiable physics-based rendering could be explored in constraint environments.

To deal with articulated objects, part poses may be determined in combination with a part-based detector. Similar to contact constraints, composition constraints may be introduced or explicitly enforced. Additionally taking, for example, hand-object contacts into account would allow to extend the physical plausibility definition to these dynamic cases. By jointly considering hand and object pose estimation, as in related work on tracking, heavy occlusion of either the hand or object during manipulation may be dealt with.

For robots to co-inhabit homes and workplaces with humans, they must moreover be able to adapt to novel objects, changing object instances and interaction therewith. Especially adapting to object properties that are not passively observable call for an interactive robot vision approach, allowing the robot to autonomously explore its environment and continuously expand its object knowledge.

Appendix

A.1 Architecture Details

Table A.1.1 details the architectures for ReAgent (Chapter 6) and SporeAgent (Chapter 7) with all used layers, their input and output dimensions. Note that the initial embedding is computed for both source and target point clouds with shared weights. Also, the action embeddings and policies are computed for both rotation and translation, although the layers are given only once in Table A.1.1.

Layer	In	Out
Embeddings $\phi(X'_i)$ and $\phi(Y)$ (shared)		
Conv1d	$N \times 3$	$N \times 64$
ReLU		
Conv1d	$N \times 64$	$N \times 128$
ReLU		
Conv1d	$N \times 128$	$N \times 1024$
max	$N \times 1024$	1×1024
State S via $\phi(X'_i) \oplus \phi(Y)$		
concat	$2(1 \times 1024)$	2048
Embeddings $\phi_R(S)$ and $\phi_t(S)$		
FC	2048	512
ReLU		
FC	512	256
ReLU		
Value $\hat{v}$, using $\phi_R(S) \oplus \phi_t(S)$		
concat	2(256)	512
FC	512	256
ReLU		
FC	256	1
Policies $\pi_R(\phi_R(S))$ and $\pi_t(\phi_t(S))$		
FC	256	33
reshape	33	3×11

Layer	In	Out
Concat global and local features $\phi(X'_i) \oplus \phi(X'_i)_1$		
Conv1d	$N \times (1024 + 64 + k)$	$N \times 512$
ReLU		
Conv1d	$N \times 512$	$N \times 256$
ReLU		
Conv1d	$N \times 256$	$N \times 128$
ReLU		
Conv1d	$N \times 128$	$N \times 1$
sigmoid		
Prune $\phi(X'_i)_3$ where sigmoid $\leq 0.5 \longrightarrow M$ points		
max	$M \times 1024$	1×1024
Yields embedding $\phi(X'_i)$ with pruned outliers		

Table A.1.1: The ReAgent (Chapter 6) and SporeAgent (Chapter 7) network architectures. The right sub-table (blue) shows the outlier removal branch in SporeAgent. Note that N indicates the number of points, k the number of classes and the index 1 in $\phi(X'_i)_1$ denotes the first layer used to compute the embedding $\phi(X'_i)$.

A.2 Code References

- Evaluation of Plausible Poses (Chapter 3): `github.com/dornik/plausible-poses`
- VeREFINE (Chapter 5): `github.com/dornik/verefine`
- ReAgent (Chapter 6): `github.com/dornik/reagent`
- SporeAgent (Chapter 7): `github.com/dornik/sporeagent`

List of Figures

List of Tables

Bibliography

[1] M. Lomas, R. Chevalier, E. V. Cross, R. C. Garrett, J. Hoare, and M. Kopack, "Explaining robot actions," in *ACM/IEEE Int. Conf. Human-Robot Interact.*, 2012, pp. 187–188 (cit. on pp. 1, 14, 79).

[2] M. M. De Graaf and B. F. Malle, "How people explain action (and autonomous intelligent systems should too)," in *AAAI Fall Symp. Ser.*, 2017 (cit. on pp. 1, 14, 79).

[3] M. M. de Graaf, B. F. Malle, A. Dragan, and T. Ziemke, "Explainable robotic systems," in *Companion of the ACM/IEEE Int. Conf. Human-Robot Interact.*, 2018, pp. 387–388 (cit. on pp. 1, 14, 79).

[4] M. Naseer, S. Khan, and F. Porikli, "Indoor scene understanding in 2.5/3d for autonomous agents: A survey," *IEEE access*, vol. 7, pp. 1859–1887, 2018 (cit. on p. 1).

[5] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "HERB: A home exploring robotic butler," *Auton. Robot.*, vol. 28, no. 1, p. 5, 2010 (cit. on p. 1).

[6] S. Chitta, E. G. Jones, M. Ciocarlie, and K. Hsiao, "Mobile manipulation in unstructured environments: Perception, planning, and execution," *IEEE Robot. Autom. Mag.*, vol. 19, no. 2, pp. 58–71, 2012 (cit. on p. 1).

[7] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. T. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *Conf. Robot. Learn.*, 2018, pp. 306–316 (cit. on p. 1).

[8] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Robot.: Sci. Syst.*, 2018 (cit. on pp. 2, 12, 15, 16, 20, 30, 41, 62–64, 69–73, 75, 76).

[9] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *Int. Conf. Comput. Vis.* (cit. on pp. 2, 11, 12).

[10] B. Drost, M. Ulrich, N. Navab, and S. Ilic, "Model globally, match locally: Efficient and robust 3D object recognition," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2010, pp. 998–1005 (cit. on pp. 2, 11, 13, 41).

[11] J. Vidal, C.-Y. Lin, and R. Martí, "6D pose estimation using an improved method based on point pair features," in *Int. Conf. Control, Automat. and Robot.*, 2018, pp. 405–409 (cit. on pp. 2, 11, 13, 33, 41).

[12] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 3343–3352 (cit. on pp. 2, 3, 12, 13, 20, 36, 41, 44–48, 63).

[13] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "Poserbpf: A rao–blackwellized particle filter for 6-d object pose tracking," *IEEE Trans. Robot.*, 2021 (cit. on pp. 2, 3, 69, 70, 72).

[14] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation," in *Eur. Conf. Comput. Vis.*, 2018, pp. 683–698 (cit. on pp. 3, 13, 30, 52, 62, 63, 70–72).

[15] J. Shao, Y. Jiang, G. Wang, Z. Li, and X. Ji, "Pfrl: Pose-free reinforcement learning for 6d pose estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 454–11 463 (cit. on pp. 3, 13, 50, 51, 54, 62, 70, 71).

[16] W. Kehl, F. Tombari, S. Ilic, and N. Navab, "Real-time 3d model tracking in color and depth on a single cpu core," 2017 (cit. on pp. 3, 12).

[17] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, 1992 (cit. on pp. 3, 12, 35, 44, 49, 55–57, 61).

[18] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *IEEE Int. Conf. 3-D Digital Imaging and Modeling*, 2001, pp. 145–152 (cit. on pp. 3, 12).

[19] V. Narayanan and M. Likhachev, "Perch: Perception via search for multi-object recognition and localization," in *Int. Conf. Robot. Autom.*, 2016, pp. 5052–5059 (cit. on pp. 3, 14).

[20] A. Aldoma, F. Tombari, L. Di Stefano, and M. Vincze, "A global hypotheses verification method for 3d object recognition," in *Eur. Conf. Comput. Vis.*, 2012, pp. 511–524 (cit. on pp. 3, 14).

[21] ——, "A global hypothesis verification framework for 3d object recognition in clutter," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 7, pp. 1383–1396, 2016 (cit. on pp. 3, 14).

[22] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "Cosypose: Consistent multi-view multi-object 6d pose estimation," in *Eur. Conf. Comput. Vis.*, 2020, pp. 574–591 (cit. on pp. 3, 12, 13, 70, 72, 73).

[23] C. Mitash, A. Boularias, and K. E. Bekris, "Improving 6d pose estimation of objects in clutter via physics-aware monte carlo tree search," in *Int. Conf. Robot. Autom.*, 2018, pp. 3331–3338 (cit. on pp. 3, 13–16, 35, 37, 39–42, 45–48, 65, 73).

[24] D. Bauer, T. Patten, and M. Vincze, "Verefine: Integrating object pose verification with physics-guided iterative refinement," 3, vol. 5, 2020, pp. 4289–4296 (cit. on pp. 6, 7, 21, 22, 35, 36, 38, 40, 42, 43, 47, 70–73).

[25] ——, "Sporeagent: Reinforced scene-level plausibility for object pose refinement," 2022, pp. 654–662 (cit. on pp. 6, 8, 21, 22, 26, 27, 65–67, 71, 73, 75–77).

[26] ——, "Physical plausibility of 6d pose estimates in scenes of static rigid objects," in *Eur. Conf. Comput. Vis. Workshops*, 2020, pp. 648–662 (cit. on pp. 6, 21, 27).

[27] ——, "Scene explanation through verification of stable object poses," 2020 (cit. on pp. 7, 21, 24, 25, 29, 30, 34).

[28] ——, "Reagent: Point cloud registration using imitation and reinforcement learning," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 14 586–14 594 (cit. on pp. 7, 49–52, 56, 57, 59, 61, 64).

[29] G. Papagni, D. Bauer, S. Köszegi, and M. Vincze, "A study design for evaluation of trust and understandability through interactive multi-modal explanations of robotic failure," 2021 (cit. on pp. 8, 79).

[30] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, *et al.*, "BOP: Benchmark for 6D object pose estimation," in *Eur. Conf. Comput. Vis.*, 2018, pp. 19–34 (cit. on pp. 11, 19).

[31] T. Hodaň, E. Brachmann, B. Drost, F. Michel, M Sundermeyer, J. Matas, and C. Rother, *BOP: Benchmark for 6D object pose estimation*, `https://bop.felk.cvut.cz/challenges/bop-challenge-2019/`, [Online; accessed 04-September-2019], 2019 (cit. on pp. 11, 41).

[32] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas, "BOP challenge 2020 on 6D object localization," *Eur. Conf. Comput. Vis. Workshops*, 2020 (cit. on pp. 11, 17–20, 22, 68).

[33] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas, "Detection and fine 3d pose estimation of texture-less objects in rgb-d images," in *Int. Conf. Intell. Robot. and Syst.*, 2015, pp. 4421–4428 (cit. on p. 11).

[34] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *ACCV*, 2012, pp. 548–562 (cit. on pp. 11, 15, 17, 18, 20, 22, 41, 62, 69).

[35] F. Tombari, A. Franchi, and L. Di Stefano, "Bold features to detect texture-less objects," in *Int. Conf. Comput. Vis.*, 2013, pp. 1265–1272 (cit. on p. 11).

[36] H. Cai, T. Werner, and J. Matas, "Fast detection of multiple textureless 3-d objects," in *Int. Conf. Comput. Vis. Sys.*, 2013, pp. 103–112 (cit. on p. 11).

[37] R. König and B. Drost, "A hybrid approach for 6dof pose estimation," in *Eur. Conf. Comput. Vis.*, Springer, 2020, pp. 700–706 (cit. on p. 11).

[38] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3109–3118 (cit. on p. 11).

[39] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation," in *Eur. Conf. Comput. Vis.*, 2016, pp. 205–220 (cit. on p. 11).

[40] T. Hodan, D. Barath, and J. Matas, "Epos: Estimating 6d pose of objects with symmetries," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 703–11 712 (cit. on p. 11).

[41] Y. Hu, J. Hugonot, P. Fua, and M. Salzmann, "Segmentation-driven 6d object pose estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 3385–3394 (cit. on p. 12).

[42] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 4561–4570 (cit. on p. 12).

[43] Y. Hu, P. Fua, W. Wang, and M. Salzmann, "Single-stage 6d object pose estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2930–2939 (cit. on p. 12).

[44] Y. Di, F. Manhardt, G. Wang, X. Ji, N. Navab, and F. Tombari, "So-pose: Exploiting self-occlusion for direct 6d pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 396–12 405 (cit. on p. 12).

[45] M. Sundermeyer, Z.-C. Marton, M. Durner, and R. Triebel, "Augmented autoencoders: Implicit 3d orientation learning for 6d object detection," *Int. J. Comput. Vis.*, pp. 1–16, 2019 (cit. on pp. 12, 32, 33).

[46] Z. Li, G. Wang, and X. Ji, "Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation," in *Int. Conf. Comput. Vis.*, 2019, pp. 7678–7687 (cit. on p. 12).

[47] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim, "Latent-class hough forests for 3d object detection and pose estimation," in *Eur. Conf. Comput. Vis.*, 2014, pp. 462–477 (cit. on p. 12).

[48] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, *et al.*, "Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3364–3372 (cit. on pp. 12, 15, 19, 41, 62, 69).

[49] D. Xu, D. Anguelov, and A. Jain, "Pointfusion: Deep sensor fusion for 3d bounding box estimation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 244–253 (cit. on p. 12).

[50] Y. Shi, J. Huang, X. Xu, Y. Zhang, and K. Xu, "Stablepose: Learning 6d object poses from geometrically stable patches," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 222–15 231 (cit. on p. 12).

[51] F. Manhardt, D. Arroyo, C. Rupprecht, B. Busam, T. Birdal, N. Navab, and F. Tombari, "Explaining the ambiguity of object detection and 6d pose from visual data," 2019 (cit. on p. 12).

[52] J. Park, Q.-Y. Zhou, and V. Koltun, "Colored point cloud registration revisited," in *Int. Conf. Comput. Vis.*, 2017, pp. 143–152 (cit. on p. 12).

[53] A. W. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vis. Comput.*, vol. 21, no. 13-14, pp. 1145–1153, 2003 (cit. on p. 12).

[54] S. Gold, A. Rangarajan, C.-P. Lu, S. Pappu, and E. Mjolsness, "New algorithms for 2d and 3d point matching: Pose estimation and correspondence," *Pattern Recognition*, vol. 31, no. 8, pp. 1019–1031, 1998 (cit. on p. 12).

[55] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3d point clouds and meshes: A survey from rigid to nonrigid," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 7, pp. 1199–1217, 2012 (cit. on p. 12).

[56] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-icp: A globally optimal solution to 3d icp point-set registration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 11, pp. 2241–2254, 2015 (cit. on p. 12).

[57] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Int. Conf. Robot. Autom.*, IEEE, 2009, pp. 3212–3217 (cit. on p. 12).

[58] R. C. Bolles and M. A. Fischler, "A ransac-based approach to model fitting and its application to finding cylinders in range data.," in *IJCAI*, Citeseer, vol. 1981, 1981, pp. 637–643 (cit. on p. 12).

[59] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *Eur. Conf. Comput. Vis.*, Springer, 2016, pp. 766–782 (cit. on pp. 12, 55–57, 61).

[60] H. Yang, J. Shi, and L. Carlone, "Teaser: Fast and certifiable point cloud registration," *IEEE Trans. Robot.*, 2020 (cit. on p. 12).

[61] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," in *Int. Conf. Comput. Vis.*, 2019, pp. 3523–3532 (cit. on pp. 12, 16, 17, 49, 54–57, 61, 62).

[62] ——, "Prnet: Self-supervised learning for partial-to-partial registration," in *Adv. Neural Inform. Process. Syst.*, 2019, pp. 8814–8826 (cit. on pp. 12, 13, 16, 51, 56).

[63] Z. J. Yew and G. H. Lee, "Rpm-net: Robust point matching using learned features," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 824–11 833 (cit. on pp. 12, 13, 17, 18, 51, 57).

[64] M. Saleh, S. Dehghani, B. Busam, N. Navab, and F. Tombari, "Graphite: Graph-induced feature extraction for point cloud registration," in *Int. Conf. 3D Vis.*, 2020, pp. 241–251 (cit. on p. 12).

[65] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2514–2523 (cit. on p. 12).

[66] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 7163–7172 (cit. on pp. 13, 16, 17, 54–57, 61).

[67] X. Li, J. K. Pontes, and S. Lucey, "Deterministic pointnetlk for generalized registration," *arXiv preprint arXiv:2008.09527*, 2020 (cit. on p. 13).

[68] V. Sarode, X. Li, H. Goforth, Y. Aoki, A. Dhagat, R. A. Srivatsan, S. Lucey, and H. Choset, "One framework to register them all: Pointnet encoding for point cloud alignment," *arXiv preprint arXiv:1912.05766*, 2019 (cit. on p. 13).

[69] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," *arXiv preprint arXiv:2008.09088*, 2020 (cit. on p. 13).

[70] F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6D pose refinement in RGB," in *Eur. Conf. Comput. Vis.*, 2018, pp. 800–815 (cit. on p. 13).

[71] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," in *Int. Conf. Comput. Vis.*, 2019, pp. 1941–1950 (cit. on pp. 13, 62).

[72] B. Busam, H. J. Jung, and N. Navab, "I like to move it: 6d pose estimation as an action decision process," *arXiv preprint arXiv:2009.12678*, 2020 (cit. on pp. 13, 50).

[73] A. Krull, E. Brachmann, S. Nowozin, F. Michel, J. Shotton, and C. Rother, "Poseagent: Budget-constrained 6d object pose estimation via reinforcement learning," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 6702–6710 (cit. on pp. 13, 14, 35, 39).

[74] Z. Sui, H. Chang, N. Xu, and O. C. Jenkins, "Geofusion: Geometric consistency informed scene estimation in dense clutter," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5913–5920, 2020 (cit. on pp. 13, 14).

[75] K. Wada, E. Sucar, S. James, D. Lenton, and A. J. Davison, "Morefusion: Multi-object reasoning for 6d pose estimation from volumetric fusion," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 14 540–14 549 (cit. on pp. 13, 65, 70, 72).

[76] K. Desingh, O. C. Jenkins, L. Reveret, and Z. Sui, "Physically plausible scene estimation for manipulation in clutter," in *Int. Conf. on Humanoid Robots*, 2016, pp. 1073–1080 (cit. on p. 13).

[77] Y. Sallami, S. Lemaignan, A. Clodic, and R. Alami, "Simulation-based physics reasoning for consistent scene estimation in an hri context," in *Int. Conf. Intell. Robot. and Syst.*, IEEE, 2019, pp. 7834–7841 (cit. on p. 14).

[78] P. Andras, L. Esterle, M. Guckert, T. A. Han, P. R. Lewis, K. Milanovic, T. Payne, C. Perret, J. Pitt, S. T. Powers, N. Urquhart, and S. Wells, "Trusting intelligent machines: Deepening trust within socio-technical systems," en, *IEEE Tech. Soc. Mag.*, vol. 37, no. 4, pp. 76–83, Dec. 2018, ISSN: 0278-0097, 1937-416X (cit. on pp. 14, 80).

[79] H. Lakkaraju and O. Bastani, "How do i fool you? manipulating user trust via misleading black box explanations," in *AAAI/ACM Conf. AI, Ethics, and Society*, 2020, pp. 79–85 (cit. on p. 14).

[80] D. Walton, "A dialogue system specification for explanation," *Synthese*, vol. 182, no. 3, pp. 349–374, 2011 (cit. on p. 14).

[81] P. E. Dunne, S. Doutre, and T. Bench-Capon, "Discovering inconsistency through examination dialogues," in *Proceedings of the 19th international joint conference on Artificial intelligence*, 2005, pp. 1680–1681 (cit. on p. 14).

[82] S. Anjomshoae, A. Najjar, D. Calvaresi, and K. Främling, *Explainable Agents and Robots: Results from a Systematic Literature Review*. 2019 (cit. on p. 14).

[83] A. Kanehira, K. Takemoto, S. Inayoshi, and T. Harada, "Multimodal explanations by predicting counterfactuality in videos," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019, pp. 8594–8602 (cit. on p. 14).

[84] D. Huk Park, L. Anne Hendricks, Z. Akata, A. Rohrbach, B. Schiele, T. Darrell, and M. Rohrbach, "Multimodal explanations: Justifying decisions and pointing to the evidence," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 8779–8788 (cit. on p. 14).

[85] H. Lakkaraju, E. Kamar, R. Caruana, and J. Leskovec, "Faithful and customizable explanations of black box models," in *AAAI/ACM Conf. AI, Ethics, and Society*, 2019, pp. 131–138 (cit. on p. 14).

[86] B. Lamche, U. Adıgüzel, and W. Wörndl, "Interactive explanations in mobile shopping recommender systems," in *Joint Workshop on Interfaces and Human Decis. Making in Recommender Syst.*, vol. 14, 2014 (cit. on p. 14).

[87] K. Alipour, J. P. Schulze, Y. Yao, A. Ziskind, and G. Burachas, "A study on multimodal and interactive explanations for visual question answering," *arXiv preprint arXiv:2003.00431*, 2020 (cit. on pp. 14, 79).

[88] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *Int. Conf. on Adv. Robot.*, IEEE, 2015, pp. 510–517 (cit. on pp. 16, 17).

[89] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1912–1920 (cit. on pp. 16, 55).

[90] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 652–660 (cit. on pp. 16, 54, 56).

[91] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Int. Conf. Comput. Vis.*, 2019, pp. 1588–1597 (cit. on pp. 16, 61).

[92] F. Bottarel, G. Vezzani, U. Pattacini, and L. Natale, "GRASPA 1.0: GRASPA is a robot arm grasping performance benchmark," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 836–843, 2020 (cit. on pp. 16, 17, 20, 47).

[93] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in RGB-D images," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2013, pp. 2930–2937 (cit. on p. 17).

[94] B. Drost, M. Ulrich, P. Bergmann, P. Hartinger, and C. Steger, "Introducing mvtec itodd: A dataset for 3d object recognition in industry," in *Int. Conf. Comput. Vis. Workshops*, 2017, pp. 2200–2208 (cit. on p. 19).

[95] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6d object pose estimation," in *Eur. Conf. Comput. Vis.*, Springer, 2016, pp. 606–619 (cit. on pp. 19, 22).

[96] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010 (cit. on p. 19).

[97] D. Morrison, P. Corke, and J. Leitner, "Egad! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4368–4375, 2020 (cit. on p. 20).

[98] I. A. Sucan and S. Chitta, *MoveIt*, `http://moveit.ros.org`, 2013 (cit. on pp. 20, 47).

[99] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004, ch. 2.5.2, pp. 50–51 (cit. on p. 24).

[100] P. L. Várkonyi, "Estimating part pose statistics with application to industrial parts feeding and shape design: New metrics, algorithms, simulation experiments and datasets," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 3, pp. 658–667, 2014 (cit. on p. 25).

[101] A. Del Prete, S. Tonneau, and N. Mansard, "Fast algorithms to test robust static equilibrium for legged robots," in *Int. Conf. Robot. Autom.*, 2016, pp. 1601–1607 (cit. on p. 26).

[102] K. Hauser, S. Wang, and M. R. Cutkosky, "Efficient equilibrium testing under adhesion and anisotropy using empirical contact force models," *IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1157–1169, 2018 (cit. on p. 26).

[103] Y. Or and E. Rimon, "Analytic characterization of a class of three-contact frictional equilibrium postures in three-dimensional gravitational environments," *Int. J. Robot. Res.*, vol. 29, no. 1, pp. 3–22, 2010 (cit. on p. 26).

[104] R. B. McGhee and A. A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Biosciences*, vol. 3, pp. 331–351, 1968 (cit. on p. 26).

[105] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, 469–483, 1996, ISSN: 0098-3500. [Online]. Available: `https://doi.org/10.1145/235815.235821` (cit. on p. 27).

[106] E. Cuthill and J. McKee, "Reducing the bandwidth of sparse symmetric matrices," in *Proceedings of the 24th National Conference*, 1969, pp. 157–172 (cit. on p. 31).

[107] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Int. Conf. Comput. Vis.*, 2017, pp. 1521–1529 (cit. on pp. 32, 33).

[108] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Int. Conf. Robot. Autom.*, 2011 (cit. on pp. 33, 41).

[109] M. R. Loghmani, B. Caputo, and M. Vincze, "Recognizing objects in-the-wild: Where do we stand?" In *Int. Conf. Robot. Autom.*, 2018, pp. 2170–2177 (cit. on pp. 35, 47).

[110] P. Ammirato, P. Poirson, E. Park, J. Košecká, and A. C. Berg, "A dataset for developing and benchmarking active vision," in *Int. Conf. Robot. Autom.*, 2017, pp. 1378–1385 (cit. on pp. 35, 47).

[111] F. Furrer, M. Wermelinger, H. Yoshida, F. Gramazio, M. Kohler, R. Siegwart, and M. Hutter, "Autonomous robotic stone stacking with online next best object target pose planning," in *Int. Conf. Robot. Autom.*, 2017, pp. 2350–2356 (cit. on p. 35).

[112] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2-3, pp. 235–256, 2002 (cit. on p. 38).

[113] L. Kocsis and C. Szepesvári, "Discounted ucb," in *2nd PASCAL Challenges Workshop*, 2006 (cit. on p. 40).

[114] A. Garivier and E. Moulines, "On upper-confidence bound policies for switching bandit problems," in *Int. Conf. Algorithmic Learn. Theory*, 2011, pp. 174–188 (cit. on p. 40).

[115] N. Mellado, D. Aiger, and N. J. Mitra, "Super 4pcs fast global pointcloud registration via smart indexing," in *Comput. Graph. Forum*, Wiley Online Library, vol. 33, 2014, pp. 205–215 (cit. on p. 41).

[116] D. Chetverikov, D. Svirko, D. Stepanov, and P. Krsek, "The trimmed iterative closest point algorithm," in *Object recognition supported by user interaction for service robots*, IEEE, vol. 3, 2002, pp. 545–548 (cit. on pp. 41, 45).

[117] M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Int. Conf. Comput. Vis.*, 2017, pp. 3828–3836 (cit. on pp. 41, 62, 69).

[118] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018, pp. 292–301 (cit. on pp. 41, 62, 69).

[119] S. V. Alexandrov, T. Patten, and M. Vincze, "Leveraging symmetries to improve object detection and pose estimation from range data," in *Int. Conf. Comput. Vis. Sys.*, 2019, pp. 397–407 (cit. on p. 41).

[120] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988 (cit. on pp. 41, 47).

[121] E. Coumans and Y. Bai, *Pybullet, a python module for physics simulation for games, robotics and machine learning*, `http://pybullet.org`, 2016–2019 (cit. on p. 41).

[122] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image and Vis. Comput.*, vol. 10, no. 3, pp. 145–155, 1992 (cit. on pp. 46, 47, 70–72).

[123] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," *arXiv preprint arXiv:1801.09847*, 2018 (cit. on pp. 46, 55, 70–72).

[124] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017 (cit. on p. 54).

[125] R. Jena and K. Sycara, "Loss-annealed gail for sample efficient and stable imitation learning," *arXiv preprint arXiv:2001.07798*, 2020 (cit. on p. 54).

[126] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015 (cit. on p. 54).

[127] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014 (cit. on p. 56).

[128] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019 (cit. on p. 56).

[129] T. Miller, "Explanation in artificial intelligence: Insights from the social sciences," *Artif. Intell.*, vol. 267, pp. 1–38, Feb. 2019, ISSN: 00043702 (cit. on p. 79).

[130] J. T. Cacioppo and R. E. Petty, "The need for cognition," *Journal of personality and social psychology*, vol. 42, no. 1, p. 116, 1982 (cit. on p. 81).

[131] A. R.-V. D. Pütten and N. Bock, "Development and validation of the self-efficacy in human-robot-interaction scale (se-hri)," *ACM Trans. Human-Robot Interact.*, vol. 7, no. 3, pp. 1–30, 2018 (cit. on p. 81).

[132] K. E. Schaefer, "Measuring trust in human robot interactions: Development of the "trust perception scale-hri"," in *Robust Intell. and Trust in Auton. Syst.* Springer, 2016, pp. 191–218 (cit. on p. 81).

[133] T. Kulesza, S. Stumpf, M. Burnett, S. Yang, I. Kwan, and W.-K. Wong, "Too much, too little, or just right? ways explanations impact end users' mental models," in *IEEE Symposium on Visual Languages and Human Centric Computing*, IEEE, 2013, pp. 3–10 (cit. on p. 82).

[134] A. Theodorou, R. H. Wortham, and J. J. Bryson, "Why is my robot behaving like that? designing transparency for real time inspection of autonomous robots," in *AISB Workshop on Principles of Robotics*, 2016 (cit. on p. 82).

[135] Z. C. Lipton, "The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery.," *Queue*, vol. 16, no. 3, pp. 31–57, 2018 (cit. on p. 82).

Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt wurde. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien, Dezember 2021

Dominik Bauer